



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2016년12월26일
(11) 등록번호 10-1689782
(24) 등록일자 2016년12월20일

(51) 국제특허분류(Int. Cl.)
G06F 17/30 (2006.01)
(21) 출원번호 10-2011-7031643
(22) 출원일자(국제) 2010년06월23일
심사청구일자 2015년06월22일
(85) 번역문제출일자 2011년12월30일
(65) 공개번호 10-2012-0106544
(43) 공개일자 2012년09월26일
(86) 국제출원번호 PCT/EP2010/058871
(87) 국제공개번호 WO 2011/000745
국제공개일자 2011년01월06일
(30) 우선권주장
09447028.3 2009년07월01일
유럽특허청(EPO)(EP)

(73) 특허권자
톱슨 라이선싱
프랑스 92130 이씨레플리노 루 잔다르크 1-5
(72) 발명자
반더할렌 플랭크
벨기에, 라나켄 비-3621, 파터 베르보이슬란 23
게스퀴에레 리벤
벨기에, 보르넬 비-2880, 엘리자베스트라트 28
스테스 브람
벨기에, 안트베르펜 비-2018, 밤바호스트라트 31
(74) 대리인
문경진, 김학수

(56) 선행기술조사문헌
“lifs an attribute-rich file system for
storage class memories”, Sasha Ames 외7, In
Proceedings of the 23rd IEEE / 14th NASA
Goddard Conference on Mass Storage Systems
and Technologies, 2006.05.31*
JP2008515048 A*
JP08506911 A
JP2001229056 A
*는 심사관에 의하여 인용된 문헌

전체 청구항 수 : 총 7 항

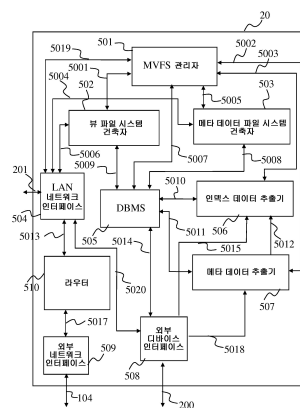
심사관 : 김병균

(54) 발명의 명칭 메타 데이터에 따라 파일 시스템의 파일들을 액세스하는 방법 및 상기 방법을 구현하는 디바이스

(57) 요약

본 발명은 파일 시스템에 저장된 데이터를 액세스하는 분야에 관한 것이고, 보다 상세하게는 파일 시스템의 데이터를 액세스하는 것을 최적화하는 것에 관한 것이다. 이러한 취지로, 본 발명은 데이터 파일 시스템의 파일들을 상기 파일들에 관련된 메타 데이터에 따라 액세스하는 방법과, 이러한 방법을 구현하는 디바이스를 제안한다.

대표도 - 도5



명세서

청구범위

청구항 1

데이터 파일 시스템(31)의 파일들을 상기 파일들에 관련된 메타 데이터에 따라 액세스하며, 파일 관리 디바이스(20)에 의해 구현되는, 파일들을 액세스하는 방법에 있어서,

상기 메타 데이터는 적어도 하나의 클라이언트 디바이스(23, 24)에 메타 데이터 파일 시스템(32)에 저장되는 것으로 표현되고, 상기 메타 데이터 파일 시스템(32)은 상기 데이터 파일 시스템(31)의 구조에 따라 구성되고,

상기 데이터 파일 시스템(31)의 적어도 하나의 제 1 파일(313)은 상기 메타 데이터 파일 시스템(32) 내의 적어도 하나의 제 1 디렉토리(324)로서 표현되고,

상기 데이터 파일 시스템의 상기 적어도 하나의 제 1 파일에 관련된 적어도 하나의 메타 데이터 속성은 상기 메타 데이터 파일 시스템(32)의 적어도 하나의 제 2 파일(325)로서 표현되고,

상기 적어도 하나의 메타 데이터 속성의 적어도 하나의 값(326)은 상기 적어도 하나의 제 2 파일(325) 내에 저장되는 것으로 표현되고,

뷰 파일 시스템(40)은 적어도 하나의 제 2 디렉토리 내의 상기 파일들에 관련된 메타 데이터에 따라 상기 데이터 파일 시스템(31)의 파일들을 구성하고, 디렉토리는 컨텐츠로서 상기 데이터 파일 시스템(31)의 상기 제 1 파일의 적어도 하나에 대한 적어도 하나의 포인터를 포함하고, 상기 컨텐츠는 적어도 하나의 메타 데이터 선택 기준에 따라 결정되고,

상기 방법은 상기 메타 데이터 파일 시스템(32)과 상기 뷰 파일 시스템(40)을 사용하여 상기 데이터 파일 시스템(32)의 상기 파일들을 액세스하는 단계를 더 포함하고, 상기 메타 데이터뿐만 아니라 상기 메타 데이터 파일 시스템과 상기 뷰 파일 시스템은 상기 데이터 파일 시스템을 액세스하기 위해 사용된 액세스 방법들을 통해 액세스되고, 상기 데이터 파일 시스템, 상기 메타 데이터 파일 시스템 및 상기 뷰 파일 시스템은 동일한 루트 파일 시스템 레벨상에 장착되는 것을 특징으로 하는, 파일들을 액세스하는 방법.

청구항 2

제 1항에 있어서, 상기 메타 데이터 파일 시스템(32)과 상기 뷰 파일 시스템(40)은 상기 데이터 파일 시스템(31)을 액세스하기 위하여 필요한 파일 시스템 액세스 방법들의 동일한 세트를 사용하여 상기 적어도 하나의 클라이언트 디바이스(23, 34)에 액세스될 수 있는 것을 특징으로 하는, 파일들을 액세스하는 방법.

청구항 3

제 1항 또는 제 2항에 있어서, 상기 적어도 하나의 메타 데이터 선택 기준은, 이름이 상기 적어도 하나의 메타 데이터 선택 기준을 식별하는 부디렉토리 내의 상기 뷰 파일 시스템 내에서 상기 적어도 하나의 클라이언트 디바이스(23, 34)에 액세스될 수 있는 적어도 하나의 구성 파일에 포함되는 것을 특징으로 하는, 파일들을 액세스하는 방법.

청구항 4

제 3항에 있어서, 상기 적어도 하나의 구성 파일은 메타 데이터 속성과 메타 데이터 속성값에 의해 표현되는 메타 데이터 선택 기준을 포함하는 것을 특징으로 하는, 파일들을 액세스하는 방법.

청구항 5

제 3항에 있어서, 상기 적어도 하나의 구성 파일은 메타 데이터 속성에 의해 표현되는 메타 데이터 정렬 기준을 포함하는 것을 특징으로 하는, 파일들을 액세스하는 방법.

청구항 6

데이터 파일 시스템(31)의 파일들을 상기 파일들에 관련된 메타 데이터에 따라 액세스하기 위한 디바이스(20)에

있어서,

상기 메타 데이터를 적어도 하나의 클라이언트 디바이스(23, 24)에 메타 데이터 파일 시스템(32)에 저장되는 것으로 표현하고, 상기 메타 데이터 파일 시스템(32)은 상기 데이터 파일 시스템(31)의 구조에 따라 구성되고,

상기 디바이스(20)는,

상기 메타 데이터 파일 시스템(32)을 통해 상기 메타 데이터에 대한 액세스를 제공하는 수단(503)으로서, 상기 데이터 파일 시스템(31)의 적어도 하나의 제 1 파일(313)은 상기 메타 데이터 파일 시스템(32) 내의 적어도 하나의 제 1 디렉토리(324)로서 표현되고, 상기 데이터 파일 시스템의 상기 적어도 하나의 제 1 파일에 관련된 적어도 하나의 메타 데이터 속성은 상기 메타 데이터 파일 시스템(32)의 적어도 하나의 제 2 파일(325)로서 표현되고, 상기 적어도 하나의 메타 데이터 속성의 적어도 하나의 값(326)은 상기 적어도 하나의 제 2 파일(325) 내에 저장되는 것으로 표현되는, 상기 메타 데이터에 대한 액세스를 제공하는 수단(503)과,

적어도 하나의 제 2 디렉토리 내의 상기 파일들에 관련된 메타 데이터에 따라 상기 데이터 파일 시스템(31)의 파일들을 구성하는 뷰 파일 시스템(40)을 통해, 상기 데이터 파일 시스템(31)의 상기 파일들에 대한 액세스를 제공하는 수단(502)으로서, 디렉토리는 콘텐츠로서 상기 데이터 파일 시스템(31) 내의 상기 제 1 파일의 적어도 하나에 대한 적어도 하나의 포인터를 포함하고, 상기 콘텐츠는 적어도 하나의 메타 데이터 선택 기준에 따라 결정되는, 상기 데이터 파일 시스템(31)의 상기 파일들에 대한 액세스를 제공하는 수단(502)을 포함하고,

상기 메타 데이터뿐만 아니라 상기 메타 데이터 파일 시스템과 상기 뷰 파일 시스템은 상기 데이터 파일 시스템을 액세스하기 위해 사용된 액세스 방법들을 통해 액세스되고, 상기 데이터 파일 시스템, 상기 메타 데이터 파일 시스템 및 상기 뷰 파일 시스템은 동일한 루트 파일 시스템 레벨상에 장착되는 것을 특징으로 하는, 파일들을 상기 파일들에 관련된 메타 데이터에 따라 액세스하기 위한 디바이스.

청구항 7

제 6항에 있어서, 상기 디바이스는, 상기 메타 데이터에 대한 액세스를 제공하는 수단(503)에 의해 사용되고 상기 뷰 파일 시스템(40)의 상기 파일들에 대한 액세스를 제공하기 위한 수단(502)에 의해 사용되는 데이터를 포함하는 데이터베이스(505)를 더 포함하는 것을 특징으로 하는, 파일들을 상기 파일들에 관련된 메타 데이터에 따라 액세스하기 위한 디바이스.

발명의 설명

기술 분야

[0001] 본 발명은 파일 시스템 내에 저장된 데이터를 액세스하는 분야에 관한 것이고, 보다 상세하게는 파일 시스템의 데이터에 대한 최적화 액세스에 관한 것이다.

배경 기술

[0002] 종래 기술에 따라, 데이터 파일들은 오디오, 비디오, 오디오 및 비디오, 정지 이미지들, 텍스트, 대화형 프로그램들, 등을 포함한다. 이들 데이터 파일들은, 데이터 파일들을 디렉토리들 및 부디렉토리들로 구성하는, 계층적으로 구성된 데이터 파일 시스템들 내에 저장된다. 수 천개의 파일들을 포함하는 대형 데이터 파일 시스템들에 있어서, 막대한 수의 파일들이 존재하여, 광대한 양의 데이터를 포함하고, 대형 데이터 파일 시스템은 곧바로 관리가 어려워진다. 대형 데이터 파일 시스템들에서 막대한 양의 파일들 및 데이터를 관리하는 방식은 메타 데이터를 통한 것이다. 메타 데이터는 데이터에 대한 데이터이다. 메타 데이터의 예들은 파일 이름, 파일 유형, 파일 길이 및 생성 날짜뿐만 아니라 저자, 텍스트 문서에서의 키 워드, 이미지의 앨범 이름, 등이다. 따라서 메타 데이터는 데이터 파일 시스템에 저장된 하나 이상의 파일(들)에 대한 정보를 제공한다. 이와 같이, 메타 데이터는 사용자(들) 또는 애플리케이션(들)이 데이터 파일 시스템 내에서 그들의 길을 찾고, 이에 액세스하는 것을 도울 수 있다. 메타 데이터가 데이터 파일 시스템의 파일들을 액세스하기 위해 사용될 때, 메타 데이터가 추출되고, 추출된 메타 데이터를 데이터 파일 시스템의 파일들에 관련시키는 색인 데이터가 생성된다. 이 후 이러한 색인 데이터는 추후의 사용을 위해 저장된다. 색인 데이터는 메타 데이터에 기초하여 파일들의 신속한 조사(lookup)를 허용한다. 색인 데이터의 생성 처리는 또한 데이터 파일 시스템의 인덱싱(indexing)으로도 불린다. 인덱싱은 따라서, 메타 데이터 및, 메타 데이터와 데이터 파일 시스템의 파일들과의 관계를 추출하고, 사용자 및/또는 애플리케이션에 의한 사용을 위해 메타 데이터 및, 데이터 파일 시스템의 파일들에 대한 메타 데이터의

관계를 저장하기 위하여, 데이터 파일 시스템을 스캐닝하는 처리이다. 대형 데이터 파일 시스템로부터 중요한 양의 메타 데이터 및 파일 관련 데이터가 추출된다. 메타 데이터를 추출하는 것과 색인 데이터의 생성은 대형 데이터 파일 시스템의 관리를 손쉽게 하기 위해서는 충분하지 않다. 네트워크 내의 독립적인 데이터 파일 시스템 각각에 대해 디바이스들을 상호연결하는 현재의 경향은 이들 디바이스들에 저장된 파일들의 관리의 복잡도를 가중시킨다. 따라서 공유된 네트워크 저장 디바이스들은 예컨대, LAN(근거리 네트워크)으로 구성된 네트워크에 연결된 디바이스들 사이에서 공유된 콘텐츠의 중앙집중된 저장을 허용하는 장점을 제공한다.

발명의 내용

해결하려는 과제

- [0003] 종래 기술에 따라, 이러한 네트워크의 디바이스들 각각은 자신의 인덱싱 시스템을 구비한다. 이러한 점은 각 클라이언트가 자신의 데이터 파일 시스템 스캐닝과 자신의 메타 데이터 및 인덱스 저장을 실행하는 것을 의미한다. 따라서, 인덱스 데이터와 메타 데이터는 이러한 네트워크의 클라이언트들 사이에서 공유될 수 없다. 이는 저장 및 처리 수요에 관해 자원들의 낭비이다.
- [0004] 종래 기술에 따라, 사용자 및/또는 애플리케이션이 한정된 메타 데이터의 생성은 메타 데이터를 관리하는 애플리케이션에 의해 제안된 메타 데이터에 제한되어, 사용자 및/또는 애플리케이션을 위한 데이터 파일 시스템의 관리력을 제한한다.
- [0005] 종래 기술에 따라, 메타 데이터는 전용 애플리케이션 프로그래밍 인터페이스(Application Programming Interface : API)를 통해 액세스되어, 액세스를 메타 데이터에, 따라서 액세스를 전용 API를 구현하는 애플리케이션에 대한 메타 데이터에 따라 데이터 파일 시스템의 파일들에 제한시킨다.
- [0006] 종래 기술에 따라, 관련 메타 데이터에 따라 데이터 파일 시스템의 배치를 구성하는 방법이 존재하지 않는다.
- [0007] 따라서, 종래 기술은 메타 데이터에 따라 파일 시스템의 파일에 대한 최적화된 액세스를 허용하지 않는다.

과제의 해결 수단

- [0008] 본 발명은 종래 기술의 불편함들 중 일부를 경감시키는 것을 목표로 한다.
- [0009] 보다 상세하게, 본 발명은 소위 말하는 MVFS(메타 데이터-뷰 파일 시스템)을 통해 메타 데이터에 따라 파일 시스템의 파일에 대한 최적화된 액세스를 허용하는데, MVFS는 이들 파일들에 관련된 메타 데이터에 따라 파일 시스템의 파일들을 액세스하는 것을 허용한다.
- [0010] 이후로 '데이터 파일 시스템'으로 언급되는 파일 시스템의 파일에 대한 액세스를 최적화하기 위하여, 본 발명은 데이터베이스 내에서 MVFS에 의한 '사적인' 액세스를 위하여 색인 데이터와 메타 데이터를 내부적으로 저장하고, 내부적으로 저장된 메타 데이터를 이후로 '메타 데이터 파일 시스템'으로 언급되는 메타 데이터 파일 시스템으로서 사용자들 및/또는 애플리케이션들에 '공개적으로' 제공한다. 사용자들 및/또는 애플리케이션들이 막대한 양의 메타 데이터를 통해 그들의 길을 브라우징하는 것을 돕기 위하여, 메타 데이터 파일 시스템의 구조는 관련된 파일 시스템의 구조를 반영한다. 덧붙여, 메타 데이터 파일 시스템과 메타 데이터 자체는 정상적인 파일 시스템 액세스 방법들을 사용하는 사용자들 및/또는 애플리케이션들에 의해 액세스될 수 있다. 데이터 파일 시스템의 파일들에 대한 액세스를 추가로 최적화하기 위하여, 사용자들 및/또는 애플리케이션들은 소위 말하는 뷰들(views)을 통해, 데이터를 특정 메타 데이터에 의한 데이터 파일 시스템으로 구성할 수 있다. 뷰의 한 예는 메타 데이터의 속성 '파일의 유형'에 기초하고, 메타 데이터 속성 값 '파일의 유형 = 음악 파일'에 대해 추가로 개선된 뷰이다. 이러한 예의 뷰는 따라서 사용자 또는 애플리케이션이 데이터 파일 시스템 내의 모든 음악 파일을 열거하는 것을 허용한다. 저장 공간을 절약하기 위하여, 그리고 또한 일관성(coherence)의 이유로 인해, 뷰는 그 뷰에 대응하는 파일들을 직접 포함하지는 않지만, 뷰는 단순히 이들 파일들에 대한 포인터들을 포함한다. 이들 포인터들은 부호 연결들(symbolic links)을 통해 구현될 수 있다. 파일들에 대한 포인터들을 사용함으로써, 어떠한 파일 복사도 필요하지 않고, 덧붙여 동일한 파일이 파일 복제 없이 다수의 뷰들에 의해 지시될 수 있다. 데이터 파일 시스템의 파일들에 대한 포인터들의 목록인 뷰의 콘텐츠는 메타 데이터 선택 기준에 따라 사용자 및/또는 애플리케이션에 의해 뷰 구성 내에서 규정될 수 있다. 메타 데이터 파일 시스템의 일부로서 사용자들 및/또는 애플리케이션들에 제공되는 메타 데이터에 대한 경우와 같이, 뷰들 또는 뷰 구성들은 이후로 '뷰 파일 시스템'으로 언급되는 뷰 파일 시스템 내의 사용자들 및/또는 애플리케이션들에 제공되어, '정상' 파일 시스템 액세스 방법들을 통해 뷰들 및 뷰 구성들에 대한 액세스를 사용자들 및/또는 애플리케이션들에 제

공한다. 메타 데이터에 대해, 뷰 구성들은 '사적인' 사용을 위해 MVFS에 의해 데이터베이스 내에 내부적으로 저장되고, 내부적으로 저장된 각 뷰 구성에 대해, 사용자들 및/또는 애플리케이션들에 의해 변경될 수 있는 '공개적인' 형태가 존재한다.

- [0011] 메타 데이터에 따라 파일 시스템의 파일들에 대한 액세스를 최적화하기 위하여, 본 발명은 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 방법을 제안하는데, 이 방법은 파일 관리 디바이스에 의해 구현되고, 이 방법은, 메타 데이터가 메타 데이터 파일 시스템에 저장되는 것으로 적어도 하나의 클라이언트 디바이스에 표현되고, 이 메타 데이터 파일 시스템이 데이터 파일 시스템의 구조에 따라 구성되고,
- [0012] 데이터 파일 시스템의 적어도 하나의 제 1 파일이 메타 데이터 파일 시스템 내의 적어도 하나의 제 1 디렉토리로서 표현되고,
- [0013] 데이터 파일 시스템의 적어도 하나의 제 1 파일에 관련된 적어도 하나의 메타 데이터 속성이 메타 데이터 파일 시스템의 적어도 하나의 제 2 파일로서 표현되고,
- [0014] 적어도 하나의 메타 데이터 속성의 적어도 하나의 값이 적어도 하나의 제 2 파일 내에 저장되는 것으로 표현되고,
- [0015] 뷰 파일 시스템은 적어도 하나의 제 2 디렉토리 내의 파일에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 구성하고, 디렉토리는 컨텐츠로서 데이터 파일 시스템의 제 1 파일의 적어도 하나에 대한 적어도 하나의 포인터를 포함하고, 컨텐츠는 적어도 하나의 메타 데이터 선택 기준에 따라 결정되고,
- [0016] 상기 방법은 메타 데이터와 뷰 파일 시스템을 사용하여 데이터 파일 시스템의 파일들을 액세스하는 단계를 더 포함하고, 메타 데이터뿐만 아니라 메타 데이터 파일 시스템과 뷰 파일 시스템은 데이터 파일 시스템을 액세스하기 위해 사용된 액세스 방법을 통해 액세스되고, 데이터 파일 시스템, 메타 데이터 파일 시스템 및 뷰 파일 시스템은 동일한 루트 파일 시스템 레벨에 장착되는 것을 특징으로 한다.
- [0017] 데이터 파일 시스템의 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 방법의 변형에 따라, 메타 데이터 파일 시스템(32)과 뷰 파일 시스템은, 데이터 파일 시스템을 액세스하기 위해 필요한 파일 시스템 액세스 방법들의 세트와 동일한 세트를 사용하는 적어도 하나의 클라이언트 디바이스에 대해 액세스 가능하다.
- [0018] 데이터 파일 시스템의 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 방법의 변형에 따라, 적어도 하나의 메타 데이터 선택 기준은 적어도 하나의 구성 파일에 포함되는데, 이러한 구성 파일은 부디렉토리 이름이 적어도 하나의 메타 데이터 선택 기준을 식별하는 부디렉토리 내의 뷰 파일 시스템의 적어도 하나의 클라이언트 디바이스에 액세스될 수 있다.
- [0019] 데이터 파일 시스템의 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 방법의 변형에 따라, 적어도 하나의 구성 파일은 메타 데이터 속성과 메타 데이터 속성 값에 의해 표현되는 메타 데이터 선택 기준을 포함한다.
- [0020] 데이터 파일 시스템의 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 방법의 변형에 따라, 적어도 하나의 구성 파일은 메타 데이터 속성에 의해 표현되는 메타 데이터 정렬 기준을 포함한다.
- [0021] 본 발명은 데이터 파일 시스템의 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 디바이스를 제안하는데, 이 디바이스는 메타 데이터 파일 시스템 내에 저장되는 것으로 적어도 하나의 클라이언트 디바이스에 메타 데이터를 표현하고, 메타 데이터 파일 시스템은 데이터 파일 시스템의 구조에 따라 구성되고, 상기 디바이스는
- [0022] 메타 데이터 파일 시스템을 통해 메타 데이터에 대한 액세스를 제공하는 수단으로서,
- [0023] 데이터 파일 시스템의 적어도 하나의 제 1 파일은 메타 데이터 파일 시스템 내의 적어도 하나의 제 1 디렉토리로서 표현되고,
- [0024] 데이터 파일 시스템의 적어도 하나의 제 1 파일에 관련된 적어도 하나의 메타 데이터 속성은 메타 데이터 파일 시스템의 적어도 하나의 제 2 파일로서 표현되고,
- [0025] 적어도 하나의 메타 데이터 속성의 적어도 하나의 값은 적어도 하나의 제 2 파일에 저장되는 것으로 표현되는,

- [0026] 메타 데이터에 대한 액세스를 제공하는 수단과,
- [0027] 적어도 하나의 제 2 디렉토리의 파일에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 구성하는 뷰 파일 시스템을 통해 데이터 파일 시스템의 파일들에 대한 액세스를 제공하는 수단으로서, 제 2 디렉토리는 콘텐츠로서 데이터 파일 시스템의 제 1 파일의 적어도 하나에 대한 적어도 하나의 포인터를 포함하고, 이러한 콘텐츠는 적어도 하나의 메타 데이터 선택 기준에 따라 결정되는, 데이터 파일 시스템의 파일들에 대한 액세스를 제공하는 수단을 포함하고,
- [0028] 메타 데이터뿐만 아니라, 메타 데이터 파일 시스템과 뷰 파일 시스템은 데이터 파일 시스템을 액세스하기 위해 사용된 액세스 방법들을 통해 액세스되고, 데이터 파일 시스템, 메타 데이터 파일 시스템 및 뷰 파일 시스템은 동일한 루트 파일 시스템 레벨에 장착되는 것을 특징으로 한다.
- [0029] 데이터 파일 시스템의 파일들에 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 디바이스의 변형에 따라, 디바이스는 메타 데이터에 대한 액세스를 제공하는 수단에 의해 사용되고, 뷰 파일 시스템의 파일들에 대한 액세스를 제공하는 수단에 의해 사용되는 데이터를 포함하는 데이터베이스를 더 포함한다.
- [0030] 본 발명의 추가 장점은 본 발명의 특징의, 비제한적인 실시예들의 설명을 통해 자명해질 것이다. 이러한 실시예들은 첨부된 도면을 참조하여 설명될 것이다.

발명의 효과

- [0031] 본 발명은 파일 시스템의 파일에 대한 최적화된 액세스를 허용한다.

도면의 간단한 설명

- [0032] 도 1은 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들에 대해 로컬 네트워크에 연결된 디바이스들에 의한 종래 기술의 액세스를 위한 예시적인 네트워크 하부구조를 도시하는 도면.
- 도 2는 본 발명과 호환되고, 로컬 네트워크에 연결된 수 개의 디바이스들을 포함하는 네트워크 하부구조에서의 본 발명의 예시적인 실시예를 도시하는 도면.
- 도 3은 메타 데이터가 본 발명의 특정 실시예에 따른 파일 시스템으로서 어떻게 제공되는 지를 도시하는 도면.
- 도 4는 뷰 데이터가 본 발명의 특정 실시예에 따른 파일 시스템으로서 어떻게 제공되는 지, 그리고 뷰 데이터가 데이터 파일 시스템의 데이터와 어떻게 관련되는 지를 도시하는 도면.
- 도 5는 본 발명의 특정 실시예에 따라 도 2의 디바이스(20)와 같은 본 발명을 구현하는 디바이스를 도시하는 도면.
- 도 6은 예컨대 본 발명의 특정 실시예에 따라 도 2의 디바이스(20)에 의해 구현되는, 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 알고리즘을 도시하는 도면.

발명을 실시하기 위한 구체적인 내용

- [0033] 도 1은 로컬 네트워크에 연결된 디바이스들에 의해, 관련된 메타 데이터에 따라 데이터 파일 시스템의 파일들에 액세스하는 종래 기술의 방법을 위한 예시적인 네트워크 하부구조를 도시한다.
- [0034] 하부구조는,
- [0035] - 저장 서버(10);
- [0036] - 저장 디바이스(11);
- [0037] - 제 1 디바이스(13);
- [0038] - 제 2 디바이스(14); 및
- [0039] - 근거리 네트워크(12)를 포함한다.
- [0040] 디바이스(13,14)와 저장 서버(10)는 각각 양방향 연결부들(102,103 및 101)을 통해 LAN(12)에 연결된다. 저장 서버(10)는 양방향 연결부(100)를 통해 저장 디바이스(11)에 연결된다. 저장 디바이스(11)는 데이터 파일 시스템으로 구성된 데이터(111)와, 메타 데이터 상의 추출된 메타 데이터 + 인덱스 데이터(112 및 113)를 포함한다.

양방향의 점선 화살표(1001 및 1002)는 디바이스(13)로부터 메타 데이터 상의 추출된 메타 데이터 + 인덱스 데이터(112)로 및 그 역방향의 그리고 디바이스(14)로부터 인덱스 데이터(113)로 및 그 역방향의 데이터 및 제어 흐름들을 각각 나타낸다.

[0041] 디바이스(13)와 디바이스(14)는 각각 저장 디바이스(11) 상의 저장 공간에 자신의 추출된 메타 데이터와 인덱스 데이터를 갖는다. 데이터(111)에 대해 디바이스(13)에 의한 메타 데이터의 추출과 인덱스 데이터의 생성은 이루어지고, 저장 공간(112)에 저장된다. 데이터(111)에 대해 디바이스(14)에 의한 메타 데이터의 추출과 인덱스 데이터의 생성은 이루어지고, 저장 공간(113)에 저장된다. 디바이스들(13 및 14) 중 어느 것도 추출된 메타 데이터와 인덱스 데이터를 공유하지 않고, 이는 저장 공간에 대해 특히 비효율적인데, 왜냐하면 상당 부분의 정보가 복제되고, 이는 데이터(111)를 위해 사용할 수 있는 저장 공간을 감소시키기 때문이다. 이는 또한 처리 수요 측면에서 특히 비효율적이다. 왜냐하면 각 디바이스가 스스로의 메타 데이터의 추출 및 인덱스 데이터의 생성을 수행해야 하기 때문이다. 따라서, 데이터(111)에 대한 변경이 이루어질 때, 이러한 변경은 각 메타 데이터/인덱스 데이터에 반영되어야 한다. 이는 변경이 각 메타 데이터/인덱스 데이터 내에 어떻게 신속하게 반영될지의 동기화 문제점의 부가를 초래한다. 덧붙여, 단일 변화는 각 메타 데이터/인덱스 데이터를 갱신하기 위하여 저장 디바이스(11)에 대한 수 개의 관독/기록 액세스들을 생성한다. 덧붙여, 메타 데이터/인덱스 데이터는 디바이스(13 및 14) 사이에서 공유되지 않아서, 각 디바이스(13 및 14)는 다른 디바이스(13 또는 14) 중 하나에 의해 이미 이루어진 메타 데이터의 추출과 인덱스 데이터의 생성으로부터 이점을 얻을 수 없다.

[0042] 상술한 종래 기술의 구현은 하나의 예시적인 종래 기술의 구현이다. 데이터 파일 시스템과 인덱스 및 메타 데이터가 저장된 저장 디바이스와 함께 집적된 단일 디바이스와 같은 다른 종래 기술의 구현들이 가능하다.

[0043] 도 2는 본 발명과 호환되는 네트워크 하부구조에서 본 발명의 예시적인 실시예를 도시한다.

[0044] 하부구조는,

[0045] - 게이트웨이(20);

[0046] - 저장 디바이스(21);

[0047] - 제 1 디바이스(23);

[0048] - 제 2 디바이스(24);

[0049] - 근거리 네트워크(12); 및

[0050] - 외부 네트워크(15)를 포함한다.

[0051] 디바이스(23, 24)와 게이트웨이(20)는 각 양방향 연결부들(202, 203 및 201)을 통해 LAN(12)에 연결된다. 디바이스(20)는 양방향 연결부(200)를 통해 저장 디바이스(21)에 연결된다. 저장 디바이스(21)는 데이터(211)와, "MVFS 데이터"(213)로 표시된 추출된 메타 데이터 + 생성된 인덱스 데이터 + 생성된 뷰들을 포함한다. 양방향의 점선 화살표(2001 및 2002)는 클라이언트 디바이스(23)로부터 추출된 메타 데이터 + 인덱스 데이터 + 생성된 뷰들(213)로 및 그 역방향의, 그리고 클라이언트 디바이스(24)로부터 동일한 추출 메타 데이터 + 생성된 인덱스 데이터 + 생성된 뷰들(213)로 및 그 역방향의 데이터 및 제어 흐름들을 각각 나타낸다. 디바이스(20)는 양방향 연결부(104)를 통해 외부 네트워크(15)에 추가로 연결되는데, 양방향 연결부(104)는 외부 네트워크(15)에 연결된 디바이스들(미도시)에 의해 제공된 추가적인 서비스들 및/또는 저장장치에 대한 액세스를 클라이언트 디바이스(23 및 24)에 제공한다.

[0052] 도 1의 종래 기술과는 대조적으로, 도 2의 클라이언트 디바이스(23 및 24)는 저장 디바이스(21) 상의 저장 공간(213)을 공유한다.

[0053] 데이터(211)로부터 메타 데이터의 추출, 인덱스 데이터의 생성, 메타 데이터 파일 시스템과 뷰 파일 시스템의 수여는 파일 관리 디바이스(20)에 의해 이루어지는데, 이것은 자원이 사용될 수 있을 때 임의의 시간에 예컨대 클라이언트 디바이스(23 및 24)의 활동이 전혀 없거나 거의 없는 저녁 시간 도중에 메타 데이터의 추출과 인덱스 데이터의 생성을 행할 수 있는 디바이스 상에서 "항상 상주"하는 장점을 갖는다.

[0054] 본 발명의 변형 실시예에 따라, 메타 데이터의 추출과 인덱스 데이터의 생성은 디바이스(20, 23 또는 24) 중 하나에 의해, 이들 디바이스들의 자원의 가용도에 따라, 평등하게 이루어진다. 이러한 변형 실시예는 각 디바이스(20, 23 또는 24)의 자원들의 가용도에 따라 메타 데이터를 추출하고 인덱스 데이터를 생성하기 위해 필요한 자원을 분산시키는 장점을 갖고, 따라서 이들 디바이스들 각각의 처리 수요들을 감소시킨다.

- [0055] 클라이언트 디바이스들(23 및 24)은 추출된 메타 데이터와 생성된 인덱스 데이터를 공유하는데, 이는 저장 공간에 관해 특히 효율적이다: 각 클라이언트 디바이스에 대해 어떠한 정보도 복제되지 않아서 종래 기술과 비교할 때 데이터(211)에 대한 저장 디바이스(21) 상에서 더 많은 공간이 사용 가능하며, 또한 이는 처리 수요들에 관해 효율적이다: 데이터(211)에 변경이 이루어질 때, 이러한 변경은 오직 하나의 메타 데이터/인덱스 데이터 저장장치(213)에서만 반영될 것이고, 이는 동기화 문제점을 회피하고, 저장 디바이스(21) 상에 저장된 메타 데이터/인덱스 데이터를 갱신하기 위하여 필요한 관독/기록 액세스들의 횟수를 줄인다. 덧붙여, 메타 데이터/인덱스 데이터가 클라이언트 디바이스(23 및 24) 사이에서 공유되어서, 클라이언트 디바이스(23 및 24)는 파일 관리 디바이스(20)에 의해, 또는 본 발명의 변형 실시예에 따라 클라이언트 디바이스(23 및 24) 중 하나에 의해 이미 이루어진 메타 데이터의 추출/인덱스 데이터의 생성으로부터 이익을 얻을 수 있다.
- [0056] 본 발명의 특정 실시예에 따라, 클라이언트 디바이스(23 및 24)는 디지털 텔레비전 및 라디오 프로그램들의 수신을 위한 디지털 셋톱박스 수신기들이다.
- [0057] 본 발명의 특정 실시예에 따라, 클라이언트 디바이스(23 및 24)는, MP3 오디오 플레이어 또는 H.264 비디오 플레이어 능력을 갖는 디바이스들과 같은 디지털 비디오 및/또는 오디오 콘텐츠를 렌더링하는 디바이스들이다.
- [0058] 본 발명의 특정 실시예에 따라, 클라이언트 디바이스(23 및 24)는 무선 연결을 통해 파일 관리 디바이스(20)에 연결된 무선 디바이스들이다.
- [0059] 도 3은 본 발명의 특정 실시예에 따라, 메타 데이터가 어떻게 파일 시스템(32)으로 제공되는 지를 도시한다.
- [0060] 수직바(30)는 데이터 파일 시스템(31)의 루트를 나타낸다. 메타 데이터 파일 시스템(32)은 내부적으로 저장된 인덱스 데이터와 메타 데이터로부터 MVFS에 의해 구성되고, 루트에 장착된 파일 시스템으로서 사용자들 및/또는 애플리케이션들에 제공된다. 데이터 파일 시스템(31)은 예컨대 도 2의 저장 디바이스(21) 상에 저장될 수 있고, 도 2의 디바이스(20)에 의해 관리된다.
- [0061] 데이터 파일 시스템(31)은 계층적인 방식으로 구성되는데, 파일 시스템 루트(30) 바로 아래에 디렉토리("a"; 311), 디렉토리("a"; 311)의 부디렉토리 ("b"; 312), 및 부디렉토리("b"; 312) 내에 파일("x.txt"; 313)을 갖는다. 파일(x.txt)은 314로 추가로 도시되고, 텍스트 콘텐츠("abcd")를 포함한다.
- [0062] 메타 데이터 파일 시스템(32)은 파일 시스템 루트(30) 바로 아래에 디렉토리("meta"; 321)를 갖는다. 디렉토리("meta")는 부디렉토리("a"; 322)를 갖고, 부디렉토리("a")는 부디렉토리("b"; 323)를 갖고, 차례로 부디렉토리("x.txt"; 324)을 갖는다.
- [0063] 메타 데이터 파일 시스템(32)은 데이터 파일 시스템(31)의 구조에 따라 이와 같이 구성되어, 데이터 파일 시스템(31)내의 각 파일은 메타 데이터 파일 시스템(32) 내의 디렉토리에 의해 표현되고, 그 디렉토리의 이름은 파일을 식별한다.
- [0064] 메타 데이터 파일 시스템(32)의 디렉토리("a"; 322)는 데이터 파일 시스템(31)의 디렉토리("a"; 311)에 대응한다. 메타 데이터 파일 시스템(32)의 디렉토리("b")는 데이터 파일 시스템(31)의 디렉토리("b"; 312)에 대응한다. 메타 데이터 파일 시스템(32)의 부디렉토리("x.txt"; 324)는 데이터 파일 시스템(31)의 파일("x.txt"; 313)에 대응한다.
- [0065] 이러한 메타 데이터 파일 시스템(32)에 있어서, 데이터 파일 시스템 내의 파일에 관련된 메타 데이터의 속성은 메타 데이터 파일 시스템의 디렉토리들 내의 하나 이상의 파일들로서 표현된다. 예컨대, 데이터 파일 시스템(31)의 파일("x.txt"; 313)은 관련된 메타 데이터 "파일 유형"을 갖고, 따라서 이러한 메타 데이터는 메타 데이터 파일 시스템(32)의 디렉토리("x.txt"; 324) 내의 파일("filetype"; 325)에 의해 표현된다. 따라서, 메타 데이터 파일 시스템(32) 내의 파일은 메타 데이터 속성을 표현한다.
- [0066] 이러한 메타 데이터 파일 시스템(32)에 있어서, 데이터 파일 시스템(31) 내의 파일에 관련된 메타 데이터 속성 값은 메타 데이터 파일 내의 값으로 표현된다. 예컨대, 파일("x.txt"; 313)에 대한 메타 데이터 속성("filetype")의 메타 데이터 값은 "textfile"이다. 따라서, 메타 데이터 속성("filetype")의 메타 데이터 값은 도 3에서 326으로 도시된, 메타 데이터 파일 시스템의 디렉토리("x.txt") 아래의 파일("filetype"; 325)의 텍스트 콘텐츠("textfile")로 표현된다.
- [0067] 이러한 방식으로, 메타 데이터 파일 시스템(32)은 사용자 및/또는 애플리케이션에 표준 계층의 파일 시스템으로 표현되고, 사용자 및/또는 애플리케이션을 위해 메타 데이터 파일 시스템(32)을 액세스하는 것은, 디렉토리의 변경, 디렉토리의 콘텐츠의 열거, 및 파일들의 콘텐츠의 관독과 같은, 데이터 파일 시스템(31)을 액세스하기 위

한 액세스 방법 이외에 다른 어떠한 액세스 방법도 필요로 하지 않는다.

- [0068] 본 발명의 특정 실시예에 따라, 사용자들 및/또는 애플리케이션들은 메타 데이터 파일 시스템(32)에 대한 제한 없는 관독/기록 액세스를 제공받는다. 이는 사용자들 및/또는 애플리케이션들에 메타 데이터 파일 시스템(32)에 관한 최대 레벨의 표현력을 허용한다.
- [0069] 또 다른 변형 실시예에 따라, 사용자들 및/또는 애플리케이션들의 관독/기록 액세스는 사용자 및/또는 애플리케이션마다 제한되어, 예컨대 메타 데이터 파일 시스템의 관리자는 제한 없는 액세스 권한을 갖는 반면, 일반 사용자는 자신이 소유한 데이터 파일 시스템(31) 내의 데이터에 관련된 메타 데이터의 부가에 제한된 액세스를 갖는, 액세스 권한의 차별을 허용한다.
- [0070] 도 4는 본 발명의 특정 실시예에 따라, 뷰들이 어떻게 MVFS의 뷰 파일 시스템(40) 내에서 생성되고, 제공되어, 유지되는지, 그리고 뷰들이 어떻게 데이터 파일 시스템(31)에 관련되는지를 도시한다.
- [0071] 도 3의 데이터 파일 시스템(31)은 도 4의 상부에 도시되었다. 파일 시스템 루트에 탑재되는 것은 내부적으로 저장된 뷰 데이터로부터 MVFS에 의해 구성되는 뷰 파일 시스템(40)이다.
- [0072] 이러한 방식으로, 뷰 파일 시스템(40)은 사용자 및/또는 애플리케이션에 표준 계층의 파일 시스템으로서 표현되고, 사용자 및/또는 애플리케이션을 위해 뷰 파일 시스템(40)을 액세스하는 것은 디렉토리의 변경, 디렉토리의 콘텐츠의 열거, 및 파일들의 콘텐츠의 관독과 같은, 데이터 파일 시스템(31)을 액세스하기 위한 액세스 방법 이외에 다른 어떠한 액세스 방법도 필요로 하지 않는다.
- [0073] 뷰 파일 시스템(40)의 베이스 디렉토리는 디렉토리("view"; 401)이다. 디렉토리("view"; 401) 아래에 디렉토리("config")가 있고, 이 디렉토리("config")는 두 개의 부디렉토리 즉, 디렉토리("file-by-type"; 403)와 디렉토리("textfiles"; 406)를 갖는다. 이들 두 디렉토리들은 파일들("cond.ini" 및 "group.ini"), 즉 디렉토리("file-by-type"; 403)에 대해 각각 404와 405, 및 디렉토리("textfiles"; 406)에 대해 각각 407과 408을 갖는다.
- [0074] 디렉토리("view"; 401) 바로 아래에는 또한 부디렉토리("textfile"; 410)를 갖는 디렉토리("file-by-type"; 409)가 존재한다. 부디렉토리("type-text"; 410)는 데이터 파일 시스템(31)의 파일("x.txt"; 313)을 가르키고, 점선 화살표(41)로 도시된, 기호 연결("a/b/x.txt"; 411)을 포함한다.
- [0075] 디렉토리("view"; 401) 바로 아래에는 또한 디렉토리("textfile"; 412)를 추가로 갖고, 이 디렉토리("textfile"; 412)는 데이터 파일 시스템(31)의 파일("x.txt"; 313)을 가르키고, 점선 화살표(41)로 도시된, 연결("a/b/x.txt")을 포함한다.
- [0076] 뷰 파일 시스템(40)은 구성 부분과 뷰 부분을 포함한다. 구성 부분은 사용자들 및/또는 애플리케이션들이 뷰의 콘텐츠를 규정하는 것을 허용하고, 구성부분은 사용자 및/또는 애플리케이션에 의해 관독 가능하고 기록 가능하다. 뷰 부분은 사용자 및/또는 애플리케이션을 위해 관독 전용이다.
- [0077] 이러한 특정 실시예에 있어서, 구성 부분은 디렉토리("config"; 402)와 그 부디렉토리들(403, 406)과 파일들(404, 405; 407, 408)에 의해 도시된다. 뷰 파일 시스템(40)은 뷰 부분을 더 포함하고, 뷰 부분은 부디렉토리(410; "textfile")와 연결(411; "a/b/x.txt")을 갖는 디렉토리(409; "files-by-type")와, 디렉토리(412; "textfiles") 및 연결(413; "a/b/x.txt")에 의해 도시된다.
- [0078] 사용자 및/또는 애플리케이션이, 파일들이 메타 데이터 선택 기준 "메타 데이터 속성 = 파일 유형"에 따라 구성된 데이터 파일 시스템(31)내의 파일들의 뷰를 원할 때, 사용자 및/또는 애플리케이션은 뷰 파일 시스템(40)내의 디렉토리("/view/config/file-by-type"; 403)를 생성한다. 이러한 디렉토리의 생성시에, MVFS는, "cond.ini"(404)와 "group.ini"(405)와 같은, 이러한 디렉토리 내의 뷰 구성 파일들을 생성할 것이다. 뷰 구성 파일들(404 및 405)의 콘텐츠는 사용자 및/또는 애플리케이션을 위해 기록 가능하지만, 이들 파일들은 사용자 및/또는 애플리케이션에 의해 삭제될 수 없다. 파일("cond.ini"; 404)은 사용자들 및/또는 애플리케이션들이, 임의의 파일 연결들이 뷰 내에서 출현하기 위하여 만족될 메타 데이터 선택 기준을 규정하는 것을 허용한다. 파일("cond.ini"; 404)이 빈채로 남겨진다면, 어떠한 파일 연결도 뷰 내에 출현하지 않을 것이다. 파일("group.ini"; 405)은 사용자 및/또는 애플리케이션이 뷰 내에서 출현하는 연결들에 대한 그룹화 기준을 규정하는 것을 허용하고, 이러한 기준에 따라, 연결들은 뷰의 부디렉토리들로 구성될 것이다. 만약 빈채로 남겨진다면, 어떠한 그룹화도 이루어지지 않는다. /view/config/files-by-type 디렉토리(430)를 생성한 후, 사용자 및/또는 애플리케이션은 메타 데이터 선택 기준("file type = *")을 파일("cond.ini"; 404)에 기록하며,

이는 데이터 파일 시스템(31)의 파일들 중 임의의 유형들이 뷰에 제공될 것임을 의미한다. 덧붙여, 사용자 및/또는 애플리케이션이 제공된 파일들을 파일 유형에 따른 뷰로 구성하기 원한다면, 사용자 및/또는 애플리케이션은 텍스트 값("type")을 파일("group.ini"; 405)에 기록한다. 사용자 및/또는 애플리케이션의 행위들의 결과로서, MVFS는 데이터 파일 시스템(31) 내에 존재하는 파일의 유형마다 부디렉토리를 갖는, 사용자 및/또는 애플리케이션이 판독 가능한 디렉토리("files-by-type"; 409)를 생성할 것이다. 본 명세서에서 주어진 예에 따라, 데이터 파일 시스템(31) 내에서 파일의 오직 한 유형만이, 즉 유형("textfile")만이 존재한다. MVFS는 따라서 부디렉토리("textfile"; 410)를 생성한다. 파일의 유형마다의 각 특정 부디렉토리에서, 그후 MVFS는 특정 파일 유형에 대응하는 모든 파일들에 대한 연결들을 생성한다. 본 명세서에서 주어진 예에 따라, 파일 유형("textfile"), 즉 파일("x.txt"; 313)에 대응하는 데이터 파일 시스템 내의 오직 하나의 파일만이 존재한다. MVFS는 따라서 부디렉토리("textfile"; 410) 내에 연결("a/b/x.text"; 411)을 생성한다.

[0079] 이제, MVFS가 메타 데이터 검색 기준에 대응하는 데이터 파일 시스템(31) 내의 파일들을 어떻게 찾는 지를 살펴보자. 상기 예를 사용하여, MVFS는 따라서 메타 데이터 속성("filetype")을 갖는 데이터 파일 시스템(31) 내의 모든 파일들을 찾고, 이들을 메타 데이터 속성 값에 따라 구성할 필요가 있다. 이를 위해, MVFS는 메타 데이터 검색 기준에 대응하는 데이터 파일 시스템 내의 파일들을 찾기 위하여 내부적으로 저장된 메타- 및 인덱스 데이터를 사용한다. 따라서, MVFS는 모든 메타 데이터 속성("filetype")을 위한 메타- 및 인덱스 데이터를 검색한다. 발견된 각 다른 파일 유형은 그 후 뷰 파일 시스템내의 부디렉토리들로 표현된다. 그후 MVFS는 발견된 모든 특정 파일 유형들에 대한 메타- 및 인덱스 데이터를 검색하고, 부디렉토리("textfile"; 410)와 같은 특정 파일 유형에 대응하는 부디렉토리 내에서 뷰 파일 시스템(40) 내의 그 파일에 대한 포인터를 생성한다. 그후, MVFS는 특정 파일 유형(= 특정 값을 갖는 메타 데이터의 속성 값)을 갖는 메타- 및 인덱스 데이터베이스(505) 내에서 인덱싱되는 각 파일에 대한 포인터를 생성하는데, 이 포인터는 메타- 및 인덱스 데이터 내에 저장되는 경로 정보로부터 생성된다.

[0080] 제 2 예로서, 사용자 및/또는 애플리케이션이 값("textfile")을 갖는 메타 데이터 속성("filetype")을 갖는 데이터 파일 시스템(31) 내의 모든 파일들의 데이터 파일 시스템(31)의 뷰를 생성하는 것을 원한다고 가정한다. 이러한 목적을 위하여, 사용자 및/또는 애플리케이션은 뷰 파일 시스템(40) 내에 디렉토리("/view/config/textfiles"; 412)를 생성하고, 이 디렉토리상에서 MVFS는 자동적으로 파일들("cond.ini" 407; "group.ini" 408)을 생성한다. 그후, 사용자 및/또는 애플리케이션은 자동적으로 생성된 "cond.ini" 파일(407)에 메타 데이터 검색 기준("file type = textfile")을 기록한다. 이는 MVFS에 의한, 파일("x.txt"; 313)을 지시하는 연결("a/b/x.txt"; 413)의 디렉토리("textfiles"; 412)의 자동 생성을 초대한다.

[0081] 상술한 실시예들은 본 발명을 설명하는 예로서 주어졌다.

[0082] 본 발명의 특정 실시예에 따라, 순서 결정 기준을 규정하기 위한 뷰 구성 파일과 같은 추가적인 뷰 구성 파일들은 사용자 및/또는 애플리케이션에 의한 뷰의 생성시에 생성되는데, 순서 결정 기준은 오름 차순 또는 내림 차순의 알파벳 순서, 또는 메타 데이터 속성 또는 메타 데이터 속성 값에 기초한 순서 결정이고, 메타 데이터 속성 값은 파일 생성 날짜 또는 파일 생성 날짜의 특정 값, 또는 파일 생성 날짜의 특정 범위이다.

[0083] 본 명세서의 독자는 파일 시스템 이름들, 파일 디렉토리 이름들 및 파일 이름들이 예시적인 목적을 위해 선택되었고, 다른 파일 시스템 이름들, 파일 디렉토리 이름들 및 파일 이름들이 본 발명과 호환됨을 이해할 것이다.

[0084] 메타 데이터 선택 기준의 명세를 위한 다른 유형들의 구문이 가능하고 본 발명과 호환된다. 특정 실시예에 있어서, XML(Extended Markup Language)의 형태(version)가 메타 데이터 검색 기준을 규정하기 위하여 사용된다. 특정 실시예에 따라, XPath(XML 경로) 질의 언어가 메타 데이터 검색 기준을 규정하기 위하여 사용된다.

[0085] 특정 실시예에 따라, 사용자 및/또는 애플리케이션들을 위한 중간의 하드웨어- 또는 소프트웨어 모듈이 제공되고, 이는 사용자 및/또는 애플리케이션이 뷰들을 생성하는 것을 단순화시킨다. 이러한 모듈은 뷰 파일 시스템의 구현 명세들을 감추는 장점을 추가로 갖는다. 이러한 모듈은 사용자 및/또는 애플리케이션으로부터와 이들에 대한 뷰들의 생성 및 관리의 임의의 복잡성을 감추면서, 사용자 및/또는 애플리케이션과 뷰 파일 시스템을 인터페이스 연결시키고, 뷰 파일 시스템에 액세스하기 위한 표준의 파일 시스템 액세스 방법을 사용한다.

[0086] 도 3과 도 4에 대해, 다음의 사항들이 적용된다.

[0087] 본 발명의 특정 실시예에 따라, 내부적으로 저장된 메타 데이터 및/또는 뷰 구성 데이터는 관계 데이터베이스에 저장된다.

[0088] 변형 실시예에 따라, 내부적으로 저장된 메타- 및 인덱스 데이터 및/또는 뷰 구성 데이터는 해쉬 테이블에 저장

된다. 해쉬 테이블은 데이터 식별자들(키들)을 데이터 값에 대해 효율적으로 맵핑하는 소위 해쉬 기능을 사용하는 데이터 구조이다. 해쉬 기능은 키를 데이터 값이 저장되는 배열 요소의 인덱스로 변환하기 위해 사용된다. 해쉬 테이블은 데이터 조사가 테이블에 저장된 요소들의 수와 독립적인 장점을 제공한다.

- [0089] 본 발명의 특정 실시예에 따라, MVFS는 데이터 파일 시스템에 대해 사용자들 및/또는 애플리케이션들에 의해 이루어진 변경을 추적한다. MVFS는, 내부적으로 저장된 데이터와 일치하는 메타 데이터와 뷰 파일 시스템을 통해 제공된 데이터의 외부 표현을 적절하게 유지하기 위하여, 메타 데이터 파일 시스템 및 뷰 파일 시스템이 되는, 내부적으로 저장된 메타- 및 인덱싱 데이터, 그 외부 표현들을 갱신한다. 마찬가지로, MVFS는 메타 데이터 파일 시스템에 대해 사용자들 및/또는 애플리케이션들에 의한 변경을 추적하고, 필요하다면 예컨대 사용자 및/또는 애플리케이션이 새로운 메타 데이터를 추가할 때 필요한 인덱싱 데이터 및/또는 뷰 파일 시스템을 갱신한다.
- [0090] 메타 데이터 및/또는 뷰 파일 시스템은 사용자 및/또는 애플리케이션에 의해 액세스될 때만 파일 시스템으로서 제공된다. 따라서, 메타 데이터 파일 시스템 및/또는 뷰 파일 시스템은 가상적이라고 말할 수 있다. 이러한 가상화는 데이터 복제를 회피하는 장점을 갖고, 따라서 데이터 불일치를 회피한다. 메타 데이터 및/또는 뷰 파일 시스템들은 따라서 내부적으로 저장된 데이터로부터 생성된 단순한 가상 표현들이다. 사용자 및/또는 애플리케이션에 의해 메타 데이터 및/또는 뷰 파일 시스템 상에서 이루어진 상술한 모든 동작들은 MVFS에 의해 내부적으로 저장된 인덱스 데이터 및 메타 데이터 상의 판독/기록 동작들로 변환된다.
- [0091] 도 5는 본 발명의 특정 실시예에 따라, 본 발명을 구현하는 도 2의 디바이스(20)와 같은 디바이스를 도시한다.
- [0092] 디바이스(20)는 다음 요소들을 포함한다:
- [0093] - 라우터(510);
- [0094] - 도 2의 디바이스(23 및 24)와 같은 디바이스들로부터 및 이들에 언급된 세가지 파일 시스템들에 대한 액세스들과 같은 요청들 및 데이터의 송신 및 수신을 허용하는, 도 2의 LAN(12)와 같은 로컬 네트워크에 디바이스(20)의 연결을 위한 근거리 네트워크 인터페이스(504);
- [0095] - 도 2의 네트워크(15)와 같은 외부 네트워크에 디바이스(20)의 연결을 위한 네트워크 인터페이스(509); 및
- [0096] - 도 2의 저장 디바이스(21)와 같은 데이터베이스(505)로부터의 데이터 파일 시스템, 메타 데이터 파일 시스템 및 뷰 파일 시스템, 및 인덱스 데이터 및 메타 데이터의 저장을 위한 외부 디바이스의 직접 연결을 위한 인터페이스(508);
- [0097] - 파일 시스템에 관한 디바이스들로부터의 액세스들을 처리하고 요소들(502-508)의 작용을 명령하는 MVFS 관리자;
- [0098] - 뷰 파일 시스템을 구축하고 유지하기 위한 뷰 파일 시스템 건축자(builder)(502);
- [0099] - 메타 데이터 파일 시스템을 구축하고 유지하기 위한 메타 데이터 파일 시스템 건축자(503);
- [0100] - 인덱스 데이터, 메타 데이터 및 뷰 구성 데이터를 저장하기 위한 DBMS(데이터베이스 관리 시스템; 505);
- [0101] - 데이터 파일 시스템으로부터 인덱스 데이터를 추출하기 위한 인덱스 데이터 추출기(506); 및
- [0102] - 데이터 파일 시스템으로부터 메타 데이터를 추출하기 위한 메타 데이터 추출기(507).
- [0103] 도 5의 요소들은 다음과 같이 상호 연결된다.
- [0104] MVFS 관리자(501)는, 양방향 연결부(5001)를 통해 뷰 파일 시스템 건축자(502)에, 양방향 연결부(5005)를 통해 메타 데이터 파일 시스템 건축자(503)에, 양방향 연결부(5003)를 통해 인덱스 데이터 추출기(506)에, 양방향 연결부(5002)를 통해 메타 데이터 추출기(507)에 연결되어, 데이터와 명령들을 언급된 요소들(502, 503, 506 및 507)에 송신하고, 언급된 요소들(502, 503, 506 및 507)로부터 데이터 및 명령의 수신확인을 수신하는 것을 허용한다. MVFS 관리자(501)는 양방향 연결부(5007)를 통해 DBMS(505)에 추가로 연결되어, 데이터 저장과 검색을 허용한다. DBMS는 양방향 연결부(5014)를 통해 외부 디바이스 인터페이스(508)에 연결되어, 외부 저장 디바이스(21) 상의 데이터베이스로부터의 데이터를 저장 공간(213)에 저장하고, 이를 검색하는 것을 허용한다. MVFS 관리자(501)는 양방향 연결부(5019)를 통해 LAN 네트워크 인터페이스(504)에 또한 연결되어, LAN(12)에 연결된 디바이스들(23 및 24)와의 데이터, 명령들 및 명령 수신확인들을 교환하는 것을 허용한다.
- [0105] 뷰 파일 시스템 건축자(502), 메타 데이터 파일 시스템 건축자(503), 인덱스 데이터 추출기(506) 및 메타 데이터 추출기(507)는 모두 각각의 양방향 연결부(5009, 5008, 5010 및 5011)를 통해 DBMS(505)에 연결되어, 이들

요소들이 데이터를 저장하고 검색하는 것을 허용한다.

- [0106] 뷰 파일 시스템 건축자(502)와 메타 데이터 파일 시스템 건축자(503)는 각각의 양방향 연결부(5006과 5004)를 통해 LAN 네트워크 인터페이스(504)에 추가로 연결되어, 이들 요소들이 LAN(12)에 연결된 디바이스들(23 및 24)와의 데이터를 교환하는 것을 허용한다.
- [0107] 인덱스 데이터 추출기(506) 및 메타 데이터 추출기(507)는 각각의 양방향 연결부(5015와 5018)를 통해 외부 디바이스 인터페이스(508)에 연결되어, 이들이 외부 저장 디바이스(21)상에 저장된 데이터 파일 시스템(31) 내에 저장된 데이터를 액세스하는 것을 허용한다. 메타 데이터 추출기(507)와 인덱스 데이터 추출기(506)는 양방향 연결부(5012)를 통해 서로 연결되어, 이들이 메타- 및 인덱스 데이터의 추출 처리시에 통신하는 것을 허용한다.
- [0108] 사용자들 및/또는 애플리케이션들은, 양방향 연결부(5020)를 통해 서로 연결된 LAN 네트워크 인터페이스(504)와 외부 디바이스 인터페이스(508)를 통해, 외부 저장 디바이스(21) 상에 저장된 데이터 파일 시스템에 대한 판독/기록 액세스를 갖는다.
- [0109] 디바이스(20)는 또한 외부 네트워크(15) 내에서 및 LAN(12) 내에서의 디바이스들 사이에서, LAN(12)에 연결된 디바이스들(23 및/또는 24) 사이에서 패킷들을 라우팅하기 위한 라우터이기 때문에, 디바이스(20)는 각각의 양방향 연결부(5013 및 5017)를 통해 LAN 네트워크 인터페이스(504)를 외부 네트워크 인터페이스(509)와 상호연결시키는 라우터 요소(510)를 포함한다.
- [0110] 최종적으로, 디바이스(20)는 양방향 연결부(201)를 통해 LAN(12)에, 양방향 연결부(104)를 통해 외부 네트워크(15)에, 양방향 연결부(200)를 통해 외부 저장 디바이스(21)에 연결된다.
- [0111] 이들 요소들은 본 발명을 구현하기 위하여 다음과 같이 함께 동작한다. 메타 데이터 추출기(507)와 인덱스 데이터 추출기(506)는 이들이 외부 디바이스 인터페이스(508)와 각각의 연결부(5018과 5015)를 통해 액세스하는 외부 디바이스(21)상에 저장된 데이터 파일 시스템으로부터 메타 데이터 및 인덱스 데이터를 추출한다. 이들이 추출한 데이터는 DBMS(505)에 저장되고, 저장될 데이터는 메타 데이터 추출기(507)와 인덱스 데이터 추출기(506)로부터 각각의 연결부(5011과 5010)를 통해 DBMS(505)로 전달된다. DBMS(505)는 새로운 데이터를 저장하고, 또는 메타 데이터 추출기로부터(5011)와 인덱스 데이터 추출기로부터(5010) 수신하는 명령들 및 데이터에 따라 내부 데이터베이스에 저장된 기존의 데이터를 갱신한다. DBMS는 저장 디바이스(21)상의 데이터베이스의 저장을 관리하고, 따라서 데이터베이스 내의 데이터가 예컨대 전력 서지로부터 야기되는 우발적인 소거로부터 보호되는 것을 보장한다. DBMS에 의해 일단 메타 데이터와 인덱스 데이터가 저장되면, 메타 데이터 추출기(507)와 인덱스 데이터 추출기(506)는 각 연결부(5002와 5003)를 통해 MVFS 관리자(501)에 종료를 통보한다. MVFS 관리자(501)는 그 후 연결부(5005)를 통해 메타 데이터 파일 시스템 건축자(503)에 데이터베이스 내에 저장된 정보에 기초하여 메타 데이터 파일 시스템을 구축하도록 명령한다. 이러한 목적을 위하여, 메타 데이터 파일 시스템 건축자는 DBMS(505)에 의해 데이터베이스 내에 저장된 데이터를 연결부(5008)를 통해 판독한다. 메타 데이터 파일 시스템 건축자가, DBMS(505)에 의해 관리되는 데이터베이스 내에 구축될 때 메타 데이터 파일 시스템의 데이터의 저장을 포함하는, 메타 데이터 파일 시스템의 구축을 종료할 때, 메타 데이터 파일 시스템 건축자(503)는 메타 데이터 파일 시스템이 구축되었고 사용될 준비가 되었음을 연결부(5005)를 통해 MVFS 관리자(501)에 통보한다. 디바이스(20)는 이제 LAN(12)에 연결된 디바이스들(23 및/또는 24)로부터 관련된 메타 데이터에 따라 데이터 파일 시스템의 데이터를 액세스하는 것과 관련된 명령들 및 요청들을 수신할 준비가 된다. 디바이스들(23 및/또는 24)은 연결부(5004)를 경유하여 LAN 네트워크 인터페이스(504)를 통해 그리고 메타 데이터 파일 시스템 건축자(503)를 통해 메타 데이터 파일 시스템을 직접 액세스한다.
- [0112] 예컨대, LAN(12)의 디바이스들(23 및/또는 24)에 의해 이루어지는, 데이터 파일 시스템(31)의 메타 데이터 및 데이터에 대한 변경은, 데이터베이스를 적절하게 갱신하도록 DBMS(505)와 통신하는 메타 데이터 및 인덱스 데이터 추출기(507 및 508)에 의해 감시된다.
- [0113] 디바이스들(23 및/또는 24)이 명령을 뷰의 구성과 액세스성에 관련된 MVFS 관리자(501)에 송신할 때, MVFS 관리자(501)는 뷰 파일 시스템 건축자에게 수신된 명령에 따라 뷰를 구성하고 제공하도록 명령한다. 뷰 파일 시스템 건축자(502)는 연결부(5009)를 통해 DBMS를 액세스함으로써 데이터베이스 내에 뷰 구성들을 저장한다. 일단 뷰가 구성되면, 뷰 파일 시스템 건축자는 디바이스들(23 및/또는 24)와 인터페이스하여, 뷰 파일 시스템 건축자(502)를 LAN 인터페이스(504)에 연결하는 연결부(5006)를 통해, 뷰 상의 판독 요청에 응답한다.
- [0114] 상술한 바와 같은 작용은 최초로 부터("from scratch"), 즉 아직까지 어떠한 메타 데이터 및 인덱스 데이터도 추출되지 않았고, 데이터베이스는 아직 채워지지 않았으며, 메타 데이터 파일 시스템은 아직 구축되지 않은 상

태로부터이다. 독자는 일단 이들 작용이 수행되면, 디바이스(20)의 매 전원투입시에 이들이 반복될 필요는 없는데, 이는 DBMS가 외부 저장 디바이스(21)상에 이미 저장된 데이터로 데이터베이스를 채움으로써 신속한 기동을 허용하기 위하여 필요한 정보를 저장하기 때문임을 이해할 것이다. 공칭 작용에 대해, 즉 DBMS(505)에 의해 관리되는 데이터베이스가 이미 채워졌을 때, MVFS 관리자(501)는 그 임무를 수행하기 위하여 기존 데이터를 갱신하고 올바른 명령을 요소들(502-507)에 전달하는 것만을 필요로 한다. 메타- 및 인덱스 데이터를 추출하는 것과 같은 시간 소모적인 작용들에는 낮은 우선순위가 주어질 수 있어서, 이들 작용의 실행은 디바이스(20)의 다른 작용들을 방해하지 않는다. 이들 낮은 우선순위의 임무는 그 후 처리 배경에서, 예컨대 휴지 시간 동안 수행될 수 있거나, 밤시간 동안, 또는 대기 상태에서, 또는 메타- 및 인덱스 데이터에 관한 데이터가 변경될 때에만, 또는 특정 양의 데이터가 변경될 때에만, 수행되도록 프로그래밍될 수 있다.

[0115] 물론, 도 5의 실시예는 본 발명의 특정 실시예를 설명하는 예시적인 실시예이다. 본 명세서의 독자는 본 발명이 다양한 다른 실시예들로 구현될 수 있음을 이해할 것이다. 예컨대, 디바이스(20)는 라우터를 구현하고, 디바이스들(23 및/또는 24)을 외부 네트워크와 인터페이스시키는 게이트웨이가 반드시 될 필요는 없지만, 본 발명의 방법을 구현하는 LAN 내의 중앙집중된 서버일 필요는 있다. 마찬가지로, 본 발명은, 프로세서상에서 실행되는 알고리즘으로서 본 발명의 방법을 구현하기 위해 필요한 처리력을 갖거나, 또는 도 5에 기술된 요소들의 일부 또는 모두와 같은 본 발명을 구현하기 위하여 필요한 전용 하드웨어를 구비한, LAN의 임의의 디바이스에서 구현될 수 있다. 독자는 LAN(12)과 같은 유선 네트워크에 연결된 디바이스를 통해 기술된 것으로 구현될 수 있지만, 임의의 유형의 무선 네트워크상에서도 구현될 수 있음을 이해할 것이다. 독자는 또한 본 발명을 구현하는 디바이스가 도 2에 도시된 것과 같은 외부 디스크를 반드시 구비할 필요는 없고, 비휘발성 유형의 메모리를 통한 디스크가 없는 구현, 또는 디바이스(20)내에 집적된 디스크 저장장치, 또는 네트워크 LAN(12)에 직접 연결된 외부 네트워크킹된 저장 디스크와 같이 다른 변형 실시예가 본 발명과 호환되는 것을 이해할 것이다.

[0116] 특정 실시예에 따라, 본 발명은 전용 구성요소(예컨대, ASIC, FPGA 또는 VLSI)(각각, 주문형 직접 회로, 현장 프로그래밍 가능 게이트 어레이, 및 초대규모 집적 회로)로서 또는 디바이스 내에 집적된 다른 전자 구성요소로서 그 전체 하드웨어로서 또는 하드웨어와 소프트웨어의 혼합 형태로서 구현될 수 있다.

[0117] 특정 실시예에 따라, 본 발명의 방법을 구현하는 디바이스는 전용 하드웨어를 구비하거나 구비하지 않은 개인용 컴퓨터이다.

[0118] 특정 실시예에 따라, 도 5의 특정 실시예의 요소들과 같은, 본 발명에 따른 기능들의 구현을 위해 필요한 요소들은 다수의 셋톱박스들 또는 다수의 게이트웨이들, 또는 이들의 혼합과 같이 네트워크 내에서 연결된 수 개의 디바이스들에 분산될 수 있고, 이들 디바이스들 각각은 하나의 가상 디바이스로서 본 발명을 구현하기 위하여 네트워크를 통해 서로 통신한다. 본 발명의 이러한 실시예는 예컨대, 기존의 장비의 처리 성능들의 사용을 통해, 중앙집중된 서버 또는 게이트웨이를 사용하는 것을 회피하기 위해 유용할 수 있다.

[0119] 도 6은 본 발명의 특정 실시예에 따라 도 2의 디바이스(20)에 의해 구현되는, 관련 메타 데이터에 따라 데이터 파일 시스템의 파일들을 액세스하는 알고리즘을 도시한다.

[0120] 알고리즘은 실행을 위해 필요한 변수들을 초기화하는 단계(600)로부터 시작한다. 그 후, 단계(601)에서, 예컨대, 도 5의 메타 데이터 추출기(507)를 사용하여 메타 데이터가 데이터 파일 시스템(31)로부터 추출되고, 예컨대 데이터베이스(505)와 같은 데이터베이스에 저장된다. 단계(602)에서, 예컨대 인덱스 데이터 추출기(506)를 사용하여 인덱스 데이터가 데이터 파일 시스템으로부터 추출되고, 예컨대 데이터베이스(505)와 같은 데이터베이스에 저장된다. 시험 단계(603)에서, 메타 데이터 파일 시스템(32)에 대한 액세스가 사용자 및/또는 애플리케이션에 의해 요청되었는지가 확인된다. 만약 액세스가 요청되었다면, 단계(604)가 실행되는데, 이 단계에서 예컨대 메타 데이터 파일 시스템 건축자(503)를 사용하여 메타 데이터 파일 시스템(32)에 대한 액세스가 주어진다. 그 후, 알고리즘은 시험 단계(603)를 반복하고, 이는 화살표(610)에 의해 표시된다. 메타 데이터 파일 시스템(32)에 대한 액세스가 요청되지 않았다면, 시험 단계(606)가 수행되는데, 이 단계에서 뷰 파일 시스템(40)에 대한 액세스가 사용자 및/또는 애플리케이션에 의해 요청되었는지가 확인된다. 만약 액세스가 요청되었다면, 단계(607)가 수행되는데, 이 단계에서 예컨대 뷰 파일 시스템 건축자(502)를 사용하여 뷰 파일 시스템(40)에 대한 액세스가 주어진다. 그 후, 알고리즘은 시험 단계(603)를 반복하고, 이는 화살표(609)에 의해 도시된다. 뷰 파일 시스템에 대한 액세스가 요청되지 않았다면, 알고리즘은 시험 단계(603)로 되돌아가고, 이는 화살표(608)에 의해 도시된다.

[0121] 본 발명의 특정 실시예에 따라, 메타 데이터 파일 시스템에 대한 액세스를 시험하고 제공하는 단계(603-604)와 뷰 파일 시스템에 대한 액세스를 시험하고 제공하는 단계(606-607)는 병렬 방식으로 실행되어, 메타 데이터 파

일 시스템과 뷰 파일 시스템에 대한 동시 액세스를 제공한다. 특정 실시예에 따라, 메타 데이터의 추출 단계(601)는 메타 데이터 파일 시스템에 대한 액세스를 제공하는 단계(604)에서 메타 데이터가 변경될 때 실행되어, 데이터베이스(505)에 저장된 메타 데이터는 갱신된채 유지된다.

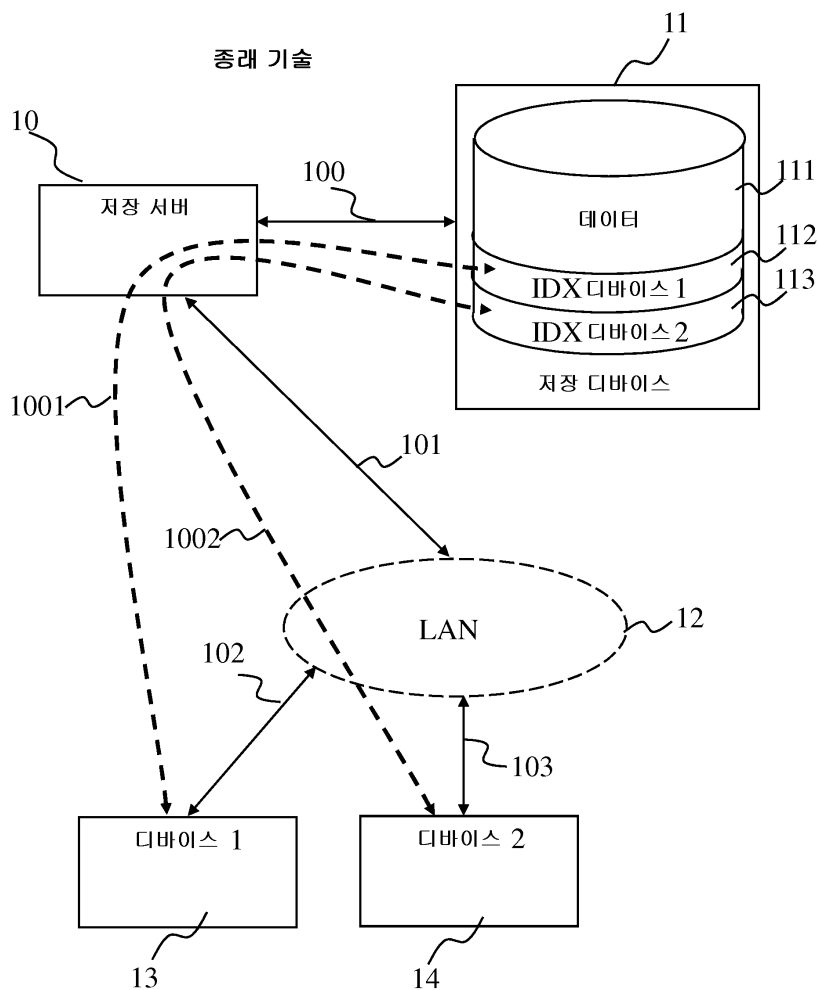
부호의 설명

[0122]

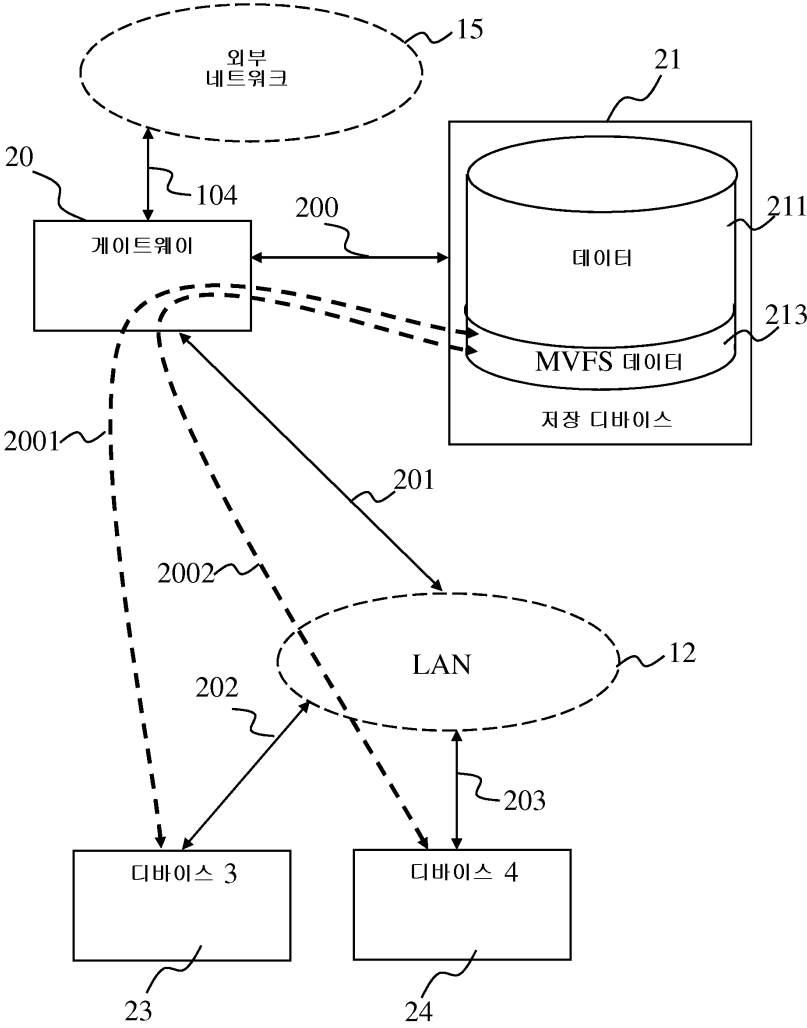
10 : 저장 서버	11,21 : 저장 디바이스
12 : 근거리 네트워크(LAN)	13,23 : 제 1 디바이스
14,24 : 제 2 디바이스	15 : 외부 네트워크
20 : 게이트웨이	30 : 파일 시스템 루트
31 : 데이터 파일 시스템	32 : 메타 데이터 파일 시스템
100,101,102,103,200,201,202,203 : 양방향 연결부	
111,211 : 파일 시스템으로 구성된 데이터	
112, 113 : 추출된 메타 데이터 + 인덱스 데이터	
213 : MVFS 데이터	

도면

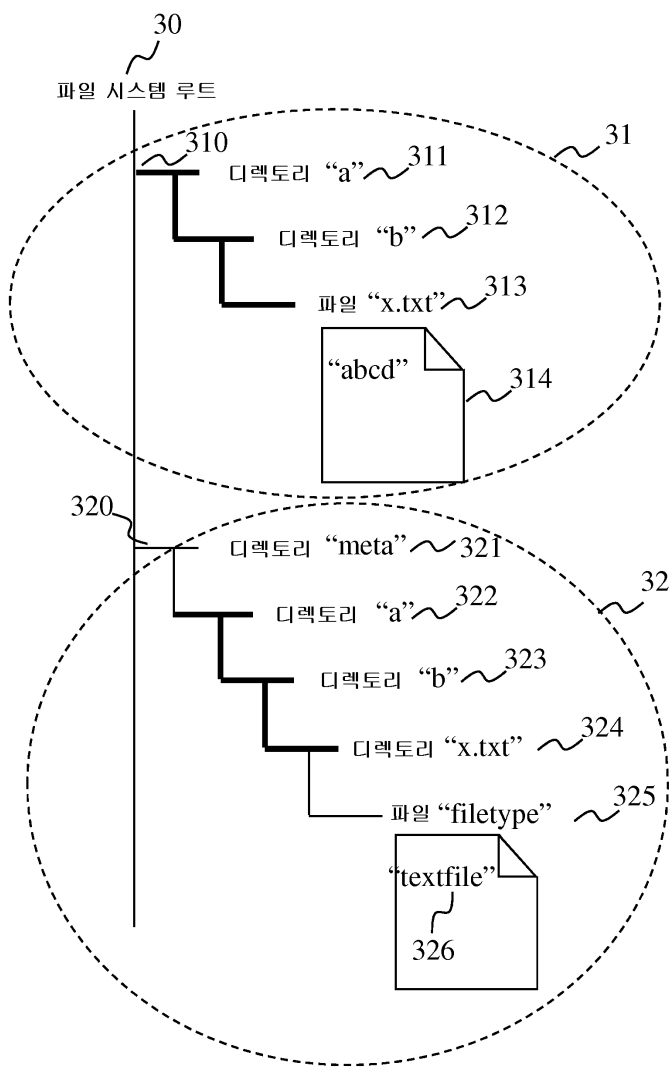
도면1



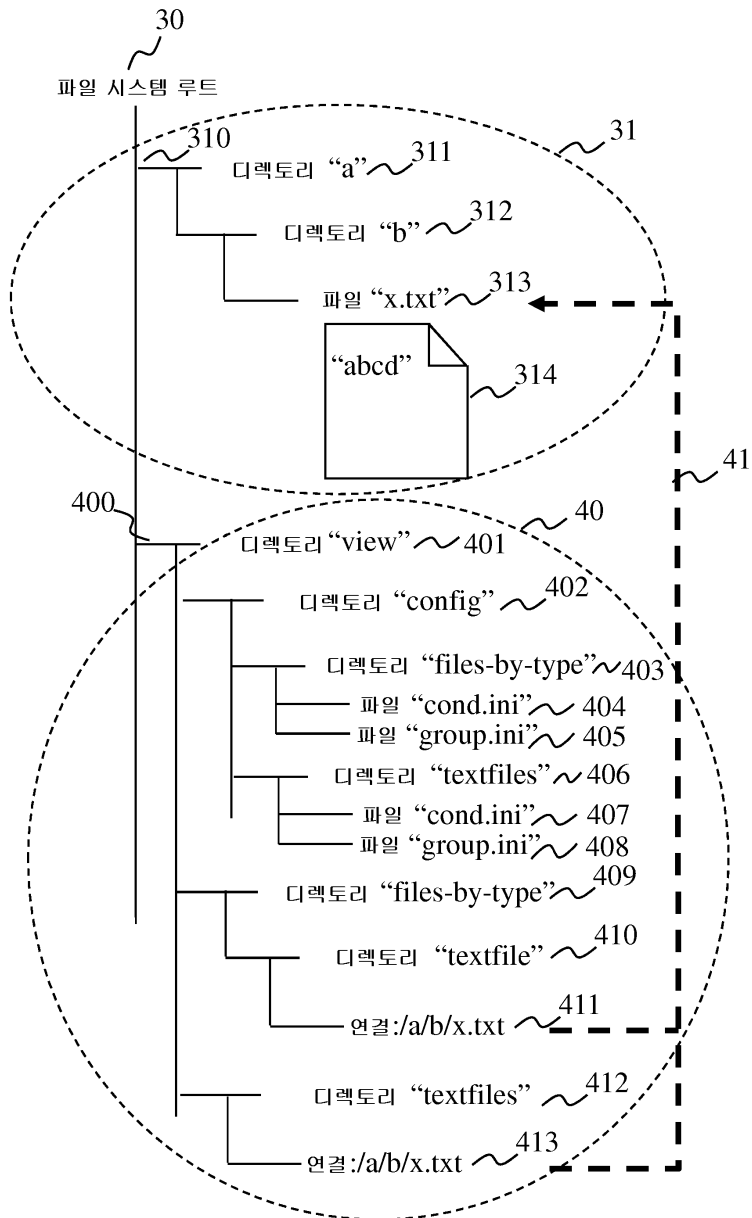
도면2



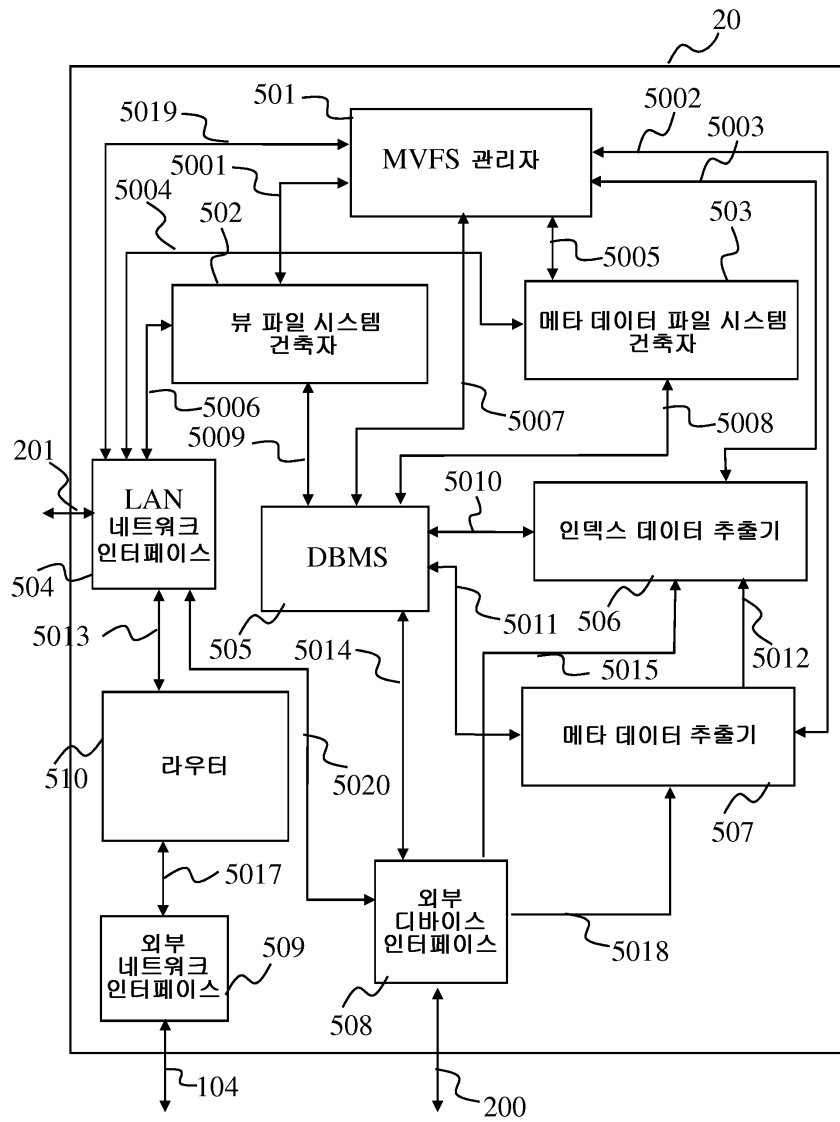
도면3



도면4



도면5



도면6

