

(19)대한민국특허청(KR)
(12) 공개특허공보(A)(51) 。 Int. Cl.⁷
G06F 12/16(11) 공개번호 10-2005-0043689
(43) 공개일자 2005년05월11일(21) 출원번호 10-2004-0089686
(22) 출원일자 2004년11월05일

(30) 우선권주장 10/702,863 2003년11월06일 미국(US)

(71) 출원인 마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이

(72) 발명자 리우,하이
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트
코포레이션 내
안토노프,로렌.
미국 98052 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트
코포레이션 내

(74) 대리인 주성민
백만기
이중희

심사청구 : 없음

(54) 이진 비교를 사용한 파일 복제 최적화

요약

클라이언트와 서버 기반의 파일의 복사본들은 그 파일에 변경이 가해지면 동기화되어 유지관리된다. 데이터는 클라이언트와 서버 모두에 알려진 이전 버전과 비교되고, 그 둘 사이의 상이점들의 고압축 표현이 생성된다. 그 다음, 이런 상이점들, 또는 "디프들(diffs)"은 전송되고, HTTP (HyperText Transport Protocol) 프로토콜의 확장들을 사용할 수 있다.

대표도

도 4

색인어

디프(diff)

명세서

도면의 간단한 설명

도 1은 본 발명의 양태들이 구현될 수 있는 컴퓨팅 환경의 예를 도시하는 블록도이다.

도 2는 본 발명에 따라 업데이트된 파일을 유지관리하는 방법의 예의 흐름도이다.

도 3은 본 발명에 따라 업데이트된 파일을 유지관리하는 방법의 다른 예의 흐름도이다.

도 4는 본 발명의 양태들의 설명에 도움을 주는 시스템의 예의 블록도를 도시한다.

도 5 및 도 6은 본 발명에 따라 업데이트된 파일을 유지관리하는 방법의 다른 예의 흐름도들이다.

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 일반적으로 컴퓨터 파일 복제 분야에 관한 것이다. 더 구체적으로, 본 발명은 이전 비교들을 사용한 파일 복제에 관한 것이다.

복제는 파일 또는 다큐먼트의 클라이언트와 서버 버전들을 동기화시켜서 데이터로의 로컬 및 원격 액세스 모두를 가능하도록 한다. 이 기능이 응용 프로그램들에게 가치있고 중요한 한편, 그것은 또한 클라이언트들과 서버 사이에 전송되어야 하는 데이터의 양 때문에 비용이 든다. 전체 파일들과 다큐먼트들은 그들의 다양한 버전들과 함께 서버에 저장되고, 서버와 그것의 클라이언트들 사이에 전송된다. 그렇게, 다수의 복제 시스템들은 그것을 전송하기 전에 데이터를 압축해서 대역폭을 절약하려고 시도한다. 그러나, 이 기존 형태의 압축은, 이 데이터의 대부분이 이전 버전의 부분으로서 전송될지라도, 전체 파일을 위한 데이터를 인코딩한다. 그러므로, 파일 또는 다큐먼트에서 데이터의 미소한 변경은, 데이터의 많은 부분이 이미 초기 수신된 버전의 형태로 목적지에 존재할지라도, 전체 파일 또는 다큐먼트가 압축되고 전송되도록 여전히 요구한다.

상술된 관점에서, 종래 기술의 한계들과 단점들을 극복하는 시스템들과 방법들에 대한 필요가 있다.

발명이 이루고자 하는 기술적 과제

본 발명은 파일에 변경이 가해지면 클라이언트와 서버 기반의 파일의 복사본들을 동기화시켜서 유지관리하는 메커니즘을 제공한다. 데이터는 클라이언트와 서버에게 모두 알려진 이전 버전과 비교되고, 그 둘 사이에 상이점들의 고압축 표현이 생성된다.

일 실시예에 따라, 기본 파일의 제1 복사본과 제2 복사본은 클라이언트에 수신되고 저장된다. 그 두 복사본들은 동일하다 - 클라이언트는 1 개의 복사본을 수신하고 그 복사본의 2개의 인스턴스들(instances)을 저장한다. 그 다음, 클라이언트는 제1 복사본에 변경을 가하고, 상이점(이전 상이점과 같은)이 변경된 제1 복사본과 제2 복사본 사이에 결정된다. 상이점은 기본 파일을 유지관리하는 서버로 전송된다. 서버는 서버의 기본 파일이 제1 디바이스에 저장된 기본 파일과 동일하면 그 상이점을 수용한다. 그렇지 않으면, 서버는 그 상이점을 거절한다.

본 발명의 양태들에 따라, 상이점이 서버에서 거절되면, 서버는 그 클라이언트에게 제2 상이점을 전송한다. 그 다음, 클라이언트는 제1 디바이스에 저장된 기본 파일의 제2 복사본에 제2 상이점을 적용한다. 이것은 서버에 있는 기본 파일에 대해 클라이언트 기본 파일을 업데이트하게 한다. 그 다음, 클라이언트는 이 업데이트된 기본 파일을 변경할 수 있고, 새로운 상이점을 생성하고, 그 새로운 상이점을 서버에게 전송한다.

본 발명의 추가 특징들과 장점들은 첨부 도면들의 참조로 진행되는 다음의 설명적 실시예들의 상세한 기재로부터 명백해질 것이다.

발명의 구성 및 작용

본 발명은 파일이 변경되면 동기화시켜서 파일의 로컬(본 명세서에서 또한 "클라이언트"로 언급됨)과 서버 기반의 복사본들을 유지관리하는 것에 관한 것이다. 본 명세서에서 설명된 시스템들과 방법들의 예는 현재 기술들보다 더 효율적이고, 동기화를 수행하는 대역폭 요구사항들과 시간 요소 모두를 최소로 유지시킨다.

데이터는 클라이언트와 서버 모두에게 알려진 이전 버전과 비교되고, 그 둘 사이의 상이점들의 고압축 표현이 생성된다. 그 다음, 이 상이점들, 또는 "디프들(diffs)"은 전송되고, HTTP (HyperText Transport Protocol) 프로토콜에 확장들을 사용할 수 있다.

도 1은 본 발명이 구현될 수 있는 적합한 컴퓨팅 환경(100)의 예를 도시한다. 컴퓨팅 시스템 환경(100)은 적합한 컴퓨팅 환경의 일 예일 뿐이고, 본 발명의 사용 또는 기능의 범위에 임의의 제한을 제안하기 위해 의도되지 않았다. 컴퓨팅 환경(100)은 운영 환경(100)의 예에 설명된 컴포넌트들 중의 임의의 것 또는 조합에 관한 임의의 의존성 또는 요구사항을 갖는 것으로 해석되어서는 안 된다.

본 발명은 다수의 다른 일반 목적 또는 특수 목적 컴퓨팅 시스템 환경들 또는 구조들과 동작한다. 본 발명과 함께 사용하기에 적합할 수 있는 잘 알려진 컴퓨팅 시스템들, 환경들, 및/또는 구조들의 예들은 개인용 PC들, 서버 컴퓨터들, 핸드헬드 또는 랩탑 디바이스들, 멀티프로세서 시스템들, 마이크로프로세서 기반 시스템들, 셋탑 박스들, 프로그램가능한 소비자 전자제품들, 통신망 PC들, 미니 컴퓨터들, 메인프레임 컴퓨터들, 상술된 시스템들 또는 디바이스들 등 중의 임의의 것을 포함하는 분산 컴퓨팅 환경들을, 하지만 거기에 국한되지 않고, 포함한다.

본 발명은 컴퓨터에 의해 실행되는 프로그램 모듈들과 같은 컴퓨터 실행가능 명령들의 일반적 컨텍스트로 기재될 수 있다. 일반적으로, 프로그램 모듈들은 특정 작업들을 수행하거나 특정 추상 데이터 타입들을 구현하는 루틴들, 프로그램들, 객체들, 컴포넌트들, 데이터 구조들 등을 포함한다. 본 발명은 또한 작업들이 통신망 또는 기타 데이터 전송 매체를 통해 연결된 원격 처리 디바이스들에 의해 수행되는 분산 컴퓨팅 환경들에서 실시될 수 있다. 분산 컴퓨팅 환경에서, 프로그램 모듈들과 기타 데이터는 메모리 저장 디바이스들을 포함하는 로컬과 원격 컴퓨터 저장 매체들 모두에 위치될 수 있다.

도 1의 참조에서, 본 발명을 구현하는 시스템의 예는 컴퓨터(110)의 형태로 일반 목적 컴퓨팅 디바이스를 포함한다. 컴퓨터(110)의 컴포넌트들은 처리 장치(120), 시스템 메모리(130), 및 처리 장치(120)에 시스템 메모리를 포함하는 다양한 시

시스템 컴포넌트들을 결합하는 시스템 버스(121)를, 하지만 거기에 국한되지 않고, 포함할 수 있다. 시스템 버스(121)는 메모리 버스 또는 메모리 제어기, 주변 버스, 및 다양한 버스 구조들 중의 임의의 것을 사용하는 로컬 버스를 포함하는 몇 가지 타입들의 버스 구조들 중의 임의의 것일 수 있다. 제한적이 아닌 단지 예로서, 그런 구조들은 산업 표준 구조 버스(Industry Standard Architecture (ISA) bus), 마이크로 채널 구조 버스(Micro Channel Architecture (MCA) bus), 진보된 ISA 버스(Enhanced ISA (EISA) bus), 비디오 전자 표준 협회 로컬 버스(Video Electronics Standards Association (VESA) local bus), 및 주변 컴포넌트 상호연결 버스(Peripheral Component Interconnect (PCI) bus)(또한 메자닌 버스라고도 알려짐)를 포함한다.

통상적으로 컴퓨터(110)는 다양한 컴퓨터 판독가능 매체들을 포함한다. 컴퓨터 판독가능 매체들은 컴퓨터(110)에 의해 액세스될 수 있는 임의의 이용가능한 매체들일 수 있고, 휘발성과 비휘발성 매체들, 분리형과 비분리형 매체들 모두를 포함한다. 제한적이 아닌 단지 예로서, 컴퓨터 판독가능 매체들은 컴퓨터 저장 매체들과 통신 매체들을 포함한다. 컴퓨터 저장 매체들은 컴퓨터 판독가능 명령들, 데이터 구조들, 프로그램 모듈들, 또는 기타 데이터와 같은 정보 저장을 위한 임의의 방법이나 기술에 구현된 휘발성과 비휘발성, 분리형과 비분리형 매체들 모두를 포함한다. 컴퓨터 저장 매체들은 RAM, ROM, EEPROM, 플래쉬 메모리 또는 기타 메모리 기술, CD-ROM, 디지털 다용도 디스크들(DVDs) 또는 기타 광 디스크 저장 장치, 자기 카세트들, 자기 테이프, 자기 디스크 저장 장치 또는 기타 자기 저장 장치 디바이스들, 또는 원하는 정보를 저장하기 위해 사용될 수 있고 컴퓨터(110)에 의해 액세스 가능한 임의의 기타 매체를, 하지만 거기에 국한되지 않고, 포함한다. 통상적으로 통신 매체들은 컴퓨터 판독가능 명령들, 데이터 구조들, 프로그램 모듈들, 또는 캐리어(carrier) 파동 또는 기타 전송 매커니즘과 같은 변조 데이터 신호의 기타 데이터를 구현하고, 임의의 정보 배달 매체들을 포함한다. "변조 데이터 신호"라는 용어는 신호에 정보를 인코딩하는 방식으로 그것의 특징들 중의 한 개 이상이 셋팅되거나 변경되는 신호를 의미한다. 제한적이 아닌 단지 예로서, 통신 매체들은 유선망이나 직접 유선 연결과 같은 유선 매체들, 그리고 음향, RF, 적외선과 같은 무선 매체들 그리고 기타 무선 매체들을 포함한다. 상술된 것들 중의 임의의 것의 조합들은 또한 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

시스템 메모리(130)는 ROM(131) 및 RAM(132)와 같은 휘발성 및/또는 비휘발성 메모리의 형태로 컴퓨터 저장 매체들을 포함한다. 스타트업(start-up)같은 때에, 컴퓨터(110) 내에 소자들 사이에 정보 전송을 돕는 기본 루틴들을 포함하는 기본 입/출력 시스템(BIOS)(133)은 통상적으로 ROM(131)에 저장된다. RAM(132)은 통상적으로 즉시 액세스 가능하고 그리고/또는 현재 처리 장치(120)에 의해 동작되고 있는 데이터 및/또는 프로그램 모듈들을 포함한다. 제한적이 아닌 단지 예로서, 도 1은 운영 체제(134), 응용 프로그램들(135), 기타 프로그램 모듈들(136), 및 프로그램 데이터(137)를 설명한다.

컴퓨터(110)는 또한 다른 분리형/비분리형, 휘발성/비휘발성 컴퓨터 저장 매체들을 포함할 수 있다. 단지 예로서, 도 1은 비휘발성, 비분리형 자기 매체들로부터 읽고 쓰는 하드 디스크 드라이브(140), 분리형, 비휘발성 자기 디스크(152)에 읽고 쓰는 자기 디스크 드라이브(151), 및 CD-ROM이나 기타 광 매체들과 같은 분리형, 비휘발성 광 디스크(156)로부터 읽고 쓰는 광 디스크 드라이브(155)를 설명한다. 운영 체제의 예에서 사용될 수 있는 다른 분리형/비분리형, 휘발성/비휘발성 컴퓨터 저장 매체들은 자기 테이프 카세트들, 플래쉬 메모리 카드들, 디지털 다용도 디스크들, 디지털 비디오 테이프, 반도체 RAM, 반도체 ROM 등을, 하지만 거기에 국한되지 않고, 포함한다. 하드 디스크 드라이브(141)는 통상적으로 인터페이스(140)와 같은 비분리형 메모리 인터페이스를 통해 시스템 버스(121)에 연결되고, 자기 디스크 드라이브(151)와 광 디스크 드라이브(155)는 통상적으로 인터페이스(150)와 같은 분리형 메모리 인터페이스에 의해 시스템 버스(121)에 연결된다.

상술되고 도 1에 설명된 드라이버들 및 그들의 연계된 컴퓨터 저장 매체들은 컴퓨터 판독가능 명령들, 데이터 구조들, 프로그램 모듈들, 및 컴퓨터(110)에 대한 기타 데이터의 저장을 제공한다. 도 1에서, 예를 들어, 하드 디스크 드라이브(141)는 운영 체제(144), 응용 프로그램들(145), 기타 프로그램 모듈들(146), 및 프로그램 데이터(147)를 저장하는 것으로 설명된다. 이런 컴포넌트들은 운영 체제(134), 응용 프로그램들(135), 기타 프로그램 모듈들(136), 및 프로그램 데이터(137)와 동일하거나 상이할 수 있음을 주목한다. 운영 체제(144), 응용 프로그램들(145), 기타 프로그램 모듈들(146), 및 프로그램 데이터(147)는, 최소한, 그들이 상이한 복사본들임을 설명하기 위해 본 명세서에서 다른 숫자들이 주어진다. 사용자는 키보드(162)와, 일반적으로 마우스, 트랙볼, 또는 터치 패드라고 언급되는 포인팅 디바이스(161)와 같은 입력 디바이스들을 통해 컴퓨터(110)로 커맨드들과 정보를 입력할 수 있다. 기타 입력 디바이스들은 (도시 안됨) 마이크로 폰, 조이스틱, 게임패드, 위성 TV, 스캐너 등을 포함할 수 있다. 이런 그리고 다른 입력 디바이스들은 종종 시스템 버스에 연결된 사용자 입력 인터페이스(160)를 통해 처리 장치(120)에 연결되지만, 병렬 포트, 게임 포트, 또는 범용 직렬 버스(USB)와 같은 다른 인터페이스와 버스 구조들에 의해 연결될 수도 있다. 모니터(191)나 다른 타입의 디스플레이 디바이스는 또한 비디오 인터페이스(190)와 같은 인터페이스를 통해 시스템 버스(121)에 연결된다. 모니터에 추가해서, 컴퓨터들은 또한 출력 주변 인터페이스(195)를 통해 연결될 수 있는 스피커들(197) 및 프린터(196)와 같은 기타 주변 출력 디바이스들을 포함할 수 있다.

컴퓨터(110)는 원격 컴퓨터(180)와 같은 한 개 이상의 원격 컴퓨터들에 논리 연결들을 사용하여 통신망 환경에서 동작할 수 있다. 원격 컴퓨터(180)는 개인용 컴퓨터, 서버, 라우터, 통신망 PC, 피어 디바이스, 또는 기타 공용 통신망 노드일 수 있고, 단지 메모리 저장 디바이스(181)만이 도 1에 설명되었지만, 통상적으로 컴퓨터(110)에 관련있는 상술된 다수 또는 전체 소자들을 포함한다. 도시된 논리 연결들은 구내 통신망(LAN)(171) 및 광역 통신망(WAN)(173)을 포함하지만, 또한 기타 통신망들을 포함할 수 있다. 그런 통신망 환경들은 사무실들, 기업 기반 컴퓨터 통신망들, 인트라넷들, 및 인터넷에서 일반적이다.

LAN 통신망 환경에서 사용될 때, 컴퓨터(110)는 통신망 인터페이스나 어댑터(170)를 통해 LAN(171)에 연결된다. WAN 통신망 환경에 사용될 때, 컴퓨터(110)는 통상적으로 인터넷과 같은 WAN(173) 상에서 통신을 개통하는 모뎀(172)이나 기타 수단을 포함한다. 내장 또는 외장일 수 있는 모뎀(172)은 사용자 입력 인터페이스(160) 또는 기타 적절한 메커니즘을 통해 시스템 버스(121)에 연결될 수 있다. 통신망 환경에서, 컴퓨터(110) 또는 그것의 일부에 관련된 도시된 프로그램 모듈들은 원격 메모리 저장 디바이스에 저장될 수 있다. 제한적이 아닌 단지 예로서, 도 1은 메모리 디바이스(181)에 있는 것으로서 원격 응용 프로그램들(185)을 설명한다. 도시된 통신망 연결들은 예일 뿐이고, 컴퓨터들 사이에 통신 연결을 개통하는 다른 수단이 사용될 수 있다.

다양한 분산 컴퓨팅 프레임워크들은 개인 컴퓨팅과 인터넷의 병합의 관점에서 개발되어져 왔고 개발되고 있다. 개인들 및 기업체 사용자들은, 컴퓨팅 활동들을 점점 더 웹 브라우저나 통신망-기반으로 만드는 응용 프로그램들과 컴퓨팅 디바이스들에 대한 자연스런 상호동작 및 웹인애이블된 인터페이스가 제공된다.

예를 들어, Microsoft® .NET 플랫폼은 서버들, 웹기반 데이터 저장과 같은 빌딩블럭(building-block) 서비스들, 및 다운로드 가능한 디바이스 소프트웨어를 포함한다. 일반적으로 말하면, .NET 플랫폼은 (1) 전체 범위의 컴퓨팅 디바이스들이 함께 동작하도록 하고 그것들 전체에 사용자 정보가 자동 업데이트되고 동기화되도록 하는 능력, (2) HTML 보다 XML을 더 사용하여 인에이블된 웹사이트들에 대한 증대된 상호작용 능력, (3) 예를 들어, 이메일 또는 Microsoft® Office®와 같은 소프트웨어와 같은 다양한 응용 프로그램들의 관리를 위한 중앙 시작점으로부터 사용자에게 제품들과 서비스들의 맞춤형 액세스와 배달의 특징이 있는 온라인 서비스들, (4) 사용자들과 디바이스들 사이에 정보 동기화를 비롯해서, 효율성과 정보로의 쉬운 액세스를 증대시키는 중앙 데이터 저장, (5) 이메일, 팩스, 및 전화들과 같은 다양한 통신 매체들을 병합하는 능력, (6) 개발자들에게, 재사용 가능한 모듈들을 생성해서, 생산성을 향상시키고 프로그램 오류들의 수를 감소시키는 능력, 및 (7) 또한 다수의 기타 플랫폼 간의 병합 특징들을 제공한다.

본 명세서에서 실시예들의 예는 컴퓨팅 디바이스에 있는 소프트웨어와 연결해서 설명되는 한편, 본 발명의 한 개 이상의 부분들은 또한, 기타 분산 컴퓨팅 프레임워크들을 비롯해서, 서비스들이 전체의 .NET 언어들과 서비스들을 통해 수행되고, 지원되고, 또는 액세스되도록, 운영 체제, API, 또는 부프로세서와 요구하는 객체 사이에 "중간자" 개체를 통해 구현될 수 있다.

도 2는 본 발명에 따라 업데이트된 파일을 유지관리하는 방법의 예의 흐름도이다. 이 실시예의 예에서, 클라이언트는 파일을 수정하고 서버에 변경들을 업로드한다. 단계(200)에서, 클라이언트는 서버에 저장된 기본 파일의 최신 버전("버전 A")의 복사본을 수신한다. 클라이언트는 버전 A를 변경하고, 단계(210)에서, 버전 A'을 생성한다. 단계(220)에서, 클라이언트는 원래 버전 A와 새 버전 A'의 복사본을 저장한다. 그러므로, 클라이언트는, 사용자가 그 파일을 업데이트할지라도, 최종 알려진 서버 상태의 복사본을 유지관리한다. 버전 A의 복사본은 단계(210) 이전 또는 단계(210) 이후에 클라이언트에 저장될 수 있음이 고려된다.

그 다음, 상이점 또는 "디프(diff)"는 버전들 A와 A'을 비교하여 단계(230)에서 생성된다. 디프는 파일의 2개의 버전들이 비교되어 더 오래된 파일에 적용되어 새로운 파일을 생성할 수 있는 압축된 디프들을 생성하는 메카니즘이다. 상이점을 찾아내는 것은 기본 형태와 수정된 형태 사이에 상이점을 결정하는 기술에 숙련자들에게 알려진 임의의 방법, 기술, 또는 시스템에 의해 수행될 수 있다. 생성된 바람직한 상이점은 이진 상이점이다. 파일은 일련의 바이트들로 고려된다. 종래 압축 알고리즘은 섀도우(shadow) 또는 기본 복사본과 수정된 복사본 사이의 상이점을 계산해서 이진 상이점을 생성하기 위해 사용된다. 그 다음, 이 상이점은 그것이 거절 또는 수용되는 서버로 전송된다. 거절은 서버의 기본 파일이 변경되었으면 발생할 수 있고, 이 경우 그 상이점은 서버에서 사용이 안된다. 임의의 상이점 엔진이나 기술은 본 발명에 따라 사용될 수 있는 것으로 사료된다. 이진 상이점 기술 사용은 본 명세서에서 예로서의 목적으로 제공된다.

더 구체적으로, 단계(240)에서, 클라이언트는 서버에게 디프를 전송한다. 기본 파일의 최신 버전은 클라이언트가 그 수정본을 만드는데에 사용한 버전 A로부터 변경되지 않았음을 확인 체크한 후에, 서버는, 단계(250)에서, 디프를 버전 A에 적용하여 파일의 새로운 최신 버전, 버전 B를 생성한다. 서버에 의해 수행된 버전들의 제깅은 도 3-6에 대해 아래 더 설명된다.

서버는 단계(260)에서 클라이언트 제공된 디프(선택적임)를 비롯하여 새로운 버전 B를 저장한다. 새 버전 B는 기본 파일의 최신 업데이트로 고려되고, 도 3-6에 대해 아래 더 설명되는 바와 같이, 디프는 원래 버전 A를 변경하는 다른 클라이언트들에 의한 사용을 위해 보관된다. 서버는 다른 클라이언트들에게 최적화된 업데이트를 제공하기 위해 디프를 선택적으로 저장한다. 복수 갱신이 되어지면, 복수 디프들이 더 오랜 버전으로부터 새로운 것을 얻기 위해 필요할 것이다. 디프는 변경을 할 의도는 아니지만 그 대신 최신 버전 B를 읽기를 원하고 이미 버전 A를 갖고 있는 클라이언트들에게 또한 유용할 수 있음을 주목한다.

단계(270)에서, 서버는 클라이언트에게 새 버전 식별자(즉, "버전 B")를 알린다. 그 다음, 클라이언트는 그것이 저장한 버전 A를 비롯해서 단계(230)에서 그것이 결정한 디프를 버리고, 클라이언트는 단계(280)에서 그것의 버전 A'을 새 버전 식별자로 표시한다. 그러므로, 클라이언트는 버전 A'을 버전 B로 재명명한다.

도 3은 업데이트된 파일을 유지관리하는 다른 방법의 예의 흐름도이다. 이 예에서, 서버는 클라이언트에게 디프 파일의 형태로 최신 변경들을 제공한다. 단계(300)에서, 파일의 버전 A를 갖는 클라이언트는 그 파일의 업데이트를 요구한다. 예를 들어, 클라이언트가 그 파일의 최신 버전을 변경하기를 바라기 때문에, 클라이언트는 그런 요구를 할 것이다. 클라이언트는 서버에게 클라이언트가 버전 A를 가졌다고 알리고, 응답으로, 서버는 단계(310)에서 A의 디프를 리턴한다. 서버는 이전 클라이언트 업데이트(즉, 도 2의 단계(260))로부터 A의 디프를 유지관리할 수 있다. 단계(320)에서, 클라이언트는 A의 디프를 그것의 저장된 버전 A에 적용하여 그 파일의 최신 버전(즉, "버전 B")을 생성한다.

도 4는 본 발명의 양태들을 설명하기에 유용한 시스템의 예의 블록도를 도시하고, 도 5와 도 6은 2명의 사용자들이 동일 기본 파일을 변경을 할 때 업데이트 파일을 유지관리하는 방법의 예의 흐름도를 도시한다. 이 예에서, 서버(400)가 기본 파일(버전 A)을 유지관리한다고 가정하면, 2개의 클라이언트들(410, 420)(본 명세서에서 클라이언트 1과 클라이언트 2로서 각각 언급됨) 모두가 동일 기본 파일을 변경하기를 원한다.

단계(500)에서, 클라이언트 1과 클라이언트 2 모두는 서버(400)로부터 기본 파일의 최신 버전("버전 A")을 요구하여 수신한다(즉, 클라이언트 1과 2는 기본 파일을 다운로드함). 클라이언트 1과 2는 기본 파일에 동시에, 또는 시간에 순차적으로, 변경을 할 수 있음이 사료된다. 그러나, 단지 1 개의 클라이언트만이 서버에 원래 기본 파일로의 그것의 변경들을 하는데에 먼저일 것이다. 이런 변경들은 원래 기본 파일에 적용된다. 그러므로, 상이점을 서버에 전송하는 제1 클라이언트는 서버에 의해 수용되는 그 상이점을 갖는다. 기본 파일에 기초한 순차적 클라이언트의 상이점은 서버에 의해 거절될 것이다. 그러므로, 원래의 기본 파일에 그것의 변경들을 전송하는 클라이언트는 나중에 그 업데이트된 기본 파일을 우선 수신해야 하고, 그 다음, 아래 더 상세히 설명하는 것처럼, 그 업데이트된 기본 파일을 변경한다.

클라이언트 1이 먼저 그것의 변경들을 하는 것으로 가정하면, 방법은 도 2에 도시된 바와 같이 단계(200) 내지 (260)에 유사하게 진행된다. 즉, 클라이언트 1은 버전 A에 그것의 변경들을 하여 단계(505)에서 버전 A'을 생성한다. 단계(510)에서, 클라이언트 1은 원래 버전 A와 새 버전 A'의 복사본을 저장한다. 버전 A의 복사본은 단계(505) 이전 또는 이후에 클라이언트 1에 저장될 수 있음이 사료된다. 그 다음, 디프(바람직하게는, 이진 디프)는 버전 A와 A'을 비교해서 단계(515)에서 생성된다.

단계(520)에서, 클라이언트 1은 서버(400)에 디프를 전송한다. 서버에 다시 변경들을 동기화시킬 때, 클라이언트는 서버가 디프 메카니즘을 지원하고, 그 다음, 원래 파일의 버전을 명시하는 버전 정보와 함께 "디프"를 업로드함을 확인한다. 기본 파일의 그 자체의 최신 버전은 클라이언트가 수정을 하는데에 사용한 버전 A로부터 변경되지 않았음을 확인 체크한 후에, 서버(400)는 버전 A의 클라이언트 1 제공된 디프를 적용하여 단계(525)에서 파일의 새로운 최신 버전, 버전 B,을 생성한다.

서버는 단계(530)에서 클라이언트 1 제공된 디프를 비롯하여 새 버전 B를 저장한다. 새 버전 B는 기본 파일의 최신 업데이트인 것으로 고려되고, 디프는 원래 버전 A를 변경시킬 다른 클라이언트들(예를 들어, 클라이언트 2)에 의해 사용되기 위해 보관된다.

도 5에 도시는 안되었지만, 단계(270, 280)에 유사하게, 서버(400)는 클라이언트 1에게 새 버전 식별자(즉, "버전 B")를 알린다. 그 다음, 클라이언트 1은 그것의 저장된 버전 A를 비롯하여 그것이 결정한 디프를 버리고, 클라이언트는 그것의 버전 A'을 새 버전 식별자로 표시한다. 그러므로, 클라이언트는 버전 A'을 버전 B로 재명명한다.

한편, 단계(535)에서 클라이언트 2는 그것이 새 버전, 버전 A'을 생성하기 위해 수신한 원래 기본 파일 버전 A를 수정한다. 단계(540)에서, 클라이언트 2는 원래 버전 A와 새 버전 A'의 복사본을 저장한다. 버전 A의 복사본은 단계(540) 이전 또는 이후에 클라이언트 2에 저장될 수 있음이 사료된다. 그 다음, 디프는 버전 A와 A'을 비교하여 단계(545)에서 생성된다.

단계(550)에서, 클라이언트 2는 그것의 버전 A의 디프를 서버(400)에 전송한다. 서버(400)는 서버가 저장하고 있는 기본 파일이 클라이언트 2가 클라이언트 2의 수정들의 기초로서 사용한 기본 파일로부터 변경했는지를 보기 위해 체크한다.

기본 파일의 서버 상태가 요구하는 클라이언트와 연계된 로컬 저장소에 저장되면, 로컬 저장소의 파일과 서버의 대응하는 파일의 상태가 비교 된다. 이 비교는 로컬 저장소에 저장된 파일의 복사본이 가장 최신 버전인지 또는 더 최신 버전이 서버에 존재하는지를 결정하기 위해 수행된다. 부연하면, 그 요구하는 클라이언트가 파일의 복사본을 얻은 최종 시간으로부터 다른 클라이언트가 그 요구하는 파일을 수정이나 업데이트한 가능성을 비교는 시사한다. 비교는 바람직하게는, 클라이언트와 서버 사이에 전체 파일의 전송을 요구하지 않고, 파일 상태를 나타내는 식별자의 전송을 포함함을 주목해야 한다. 이 방식으로, 비교는 그렇지 않으면 요구되었을 통신망 소동상태를 감소시키고, 파일의 동일 버전을 한 번 이상 전송하는 것을 피한다.

그러므로, 단계(555)에서, 서버는 기본 파일의 최신 버전이 클라이언트 2가 수정하면서 사용한 버전 A로부터 변경되지 않았음을 확인하기 위해 체크한다. 기본 파일이 변경되지 않았으면, 단계(590)에서 서버(400)는 클라이언트 2가 제공한 디프를 서버 저장된 기본 파일에 적용하여, 서버가 단계(595)에서 클라이언트 2 제공된 디프와 함께 저장하는 그 파일의 새로운 최신 버전을 생성한다. 서버(400)는 클라이언트 2에게 새 버전 식별자를 알릴 것이고, 그 다음, 클라이언트 2는 그것의 저장된 버전 A를 비롯하여 그것이 결정한 디프를 버리고, 그것의 수정된 버전 A'을 새 버전 식별자로 표시한다.

그러나, 이 예에서, 클라이언트 1은 서버(400)에 변경들을 이미 제공해서, 서버에 저장된 기본 파일은 버전 B로 변경된다. 클라이언트 2는 버전 B의 복사본을 갖지 않고, 파일의 버전 A에 그것의 수정들을 적용한다. 그러므로, 기본 파일이 변경되었기 때문에, 서버(400)는 단계(560)에서 클라이언트 2 제공된 디프를 거절하고, 서버(400)가 이전에 수신하여 저장한(단계(530)) 버전 A의 클라이언트 1 제공된 디프를 클라이언트 2에게 전송한다.

단계(565)에서, 클라이언트 2는 버전 A의 클라이언트 1 제공된 디프를 그것의 저장된 버전 A에 적용하여 파일의 최신 서버 저장된 버전을 얻는다(본 명세서에서, 버전 B). 그 다음, 클라이언트 2는 최신 버전과 그것의 수정된 버전 A' 사이에 디프를 결정하고, 단계(570, 575)에서 각각 서버(400)에 그 디프를 전송한다. 서버(400)는 단계(580)에서 그 새 디프를 그것의 저장된 최신 버전(버전 B)에 적용하여, 새로운 최신 버전(본 명세서에서, 버전 C)을 생성한다. 단계(585)에서, 서버(400)는 새로 수신된 디프를 비롯하여 새로운 최신 버전을 저장한다. 단계들(270, 280)에 유사하게, 도 5에 도시되지는 않았지만, 서버(400)는 클라이언트 2에게 새 버전 식별자(즉, "버전 C")를 알린다. 그 다음, 클라이언트는 그것의 저장된 버전을 비롯하여 그것이 결정한 디프를 버리고, 클라이언트는 그것의 버전 A'을 새 버전 식별자로 표시한다. 그러므로, 클라이언트는 버전 A'을 버전 C로 재명명한다.

서버에 수정된 버전을 자동 저장하는 대신에, 관리자와 같은 사용자는 어떻게 변경들이 병합되어야 하는지를 결정할 수 있음이 사료된다. 이것은 이전 사용자의 변경들과의 내용 마찰들을 피할 수 있다.

디프는 서버가 그 디프를 수용하기 위해 승인을 지시하기 전 또는 후에 결정될 수 있음을 주목한다. 그러므로, 증가된 효율성을 위해, 서버가 현재 최신 버전으로 유지관리하고 있는 기본 파일과 동일 버전을 클라이언트가 변경했다는 것을 서버가 지시할 때까지 클라이언트는 기다릴 것이다. 그때서야 클라이언트는 디프를 결정하고 그것을 서버에게 제공할 것이다. 바람직하게는, 서버는 그 상이점을 계산하지 않고, 그 대신 단지 그 상이점을 적용한다.

한 클라이언트가, 다른 클라이언트가 서버와 연결하여 그것의 변경들을 제공하기 전에 복수 업로드들을 하는 상황이 고려된다. 예를 들어, 원래 기본 파일이 버전 A라고 가정한다. 그 다음, 클라이언트 1은 변경들을 하고, 이런 변경들은 버전 B로서 수용된다. 클라이언트 1이 추가 변경들을 하고 그것들을 서버에 제공하면, 이 새로운 최신 버전은 버전 C로서 저장될 것이다. 바람직하게는, 서버는 버전 A와 버전 B 사이의 상이점, 및 버전 B와 버전 C 사이의 상이점을 저장한다. 그러므로,

다른 클라이언트가 그것의 변경들을 할 때, 서버는 바람직하게는 동일 메시지에 버전 A와 버전 B 사이의 상이점, 및 버전 B와 버전 C 사이의 상이점을 그 클라이언트에게 전송할 것이다. 그 다음, 클라이언트는 버전 B, 그 다음 버전 C를 재생성하고, 버전 C와 그것의 변경들 사이의 상이점을 결정하고, 이 상이점을 서버에게 제공한다.

오랜 클라이언트가 서버에 연결하여 최신 버전을 얻으면, 그것은 서버에게 어떤 버전을 가졌는지를 말하고, 저장된 디프들이 그 버전으로 되돌아가면, 적절한 디프 또는 디프들이 현재 버전 ID와 함께 리턴된다. 바람직하게는, 서버는 여전히 구 버전들에 변경들을 (즉, 기본 파일의 구 버전들에 "역행 호환(backward compatible)"하기 위함) 하고 있을 그런 클라이언트들을 병합하기 위해 처리 동안 그것이 수신하는 다양한 버전들 사이에 모든 디프들을 유지관리한다. 그러나, 불특정 시간에, 서버는 그것이 유지관리하고 있던 이전 저장된 디프들을 삭제할 수 있다. 그런 동작은, 예를 들어, 날짜 또는 저장 용량에 의해 촉구될 수 있다.

바람직하게는, HTTP(HyperText Transport Protocol)는 디프들을 전송하기 위해 사용된다. 더 구체적으로, 프로토콜 확장들은 디프가 전송되고 있거나 그렇지 않으면 구현되고 있거나 메시지 내에 병합되고 있음을 서버에게 경고하기 위해 사용될 수 있다.

HTTP는 정보가 인터넷, 인트라넷들, 및 엑스트라넷들과 같은 TCP/IP (Transmission Control Protocol/Internet Protocol) 호환성 통신망들을 통해 전송되는 표준 메카니즘으로서 발돋움하고 있다. 더 구체적으로, HTTP는 분산, 협업의 하이퍼미디어 정보 시스템들을 위한 응용 프로그램 수준의 프로토콜이다. 그것의 요구 방법들, 오류 코드들, 및 헤더들의 확장을 통해, 이름 서버들과 분산 객체 관리 시스템들과 같은 하이퍼텍스트를 위한 그것의 사용을 넘는 다수의 작업들을 위해 사용될 수 있는 일반적인고 비상태적인 프로토콜이다. 정보가 그것의 명시들에 따라 전송되기 때문에, 전송 프로토콜이라고 언급되고, 정보가 그것에 응답을 발생시키는 서버를 요구하는 클라이언트에 의해 교환되므로, 또한 요구-응답 프로토콜이라고도 언급된다. 본 명세서에서 언급된 HTTP는 일반적으로 임의의 표준 HTTP를 언급하고 웹사이트 <http://www.w3.org>에서 이용가능하다.

HTTP의 일반 사용은 마크업 언어에 따라 형식화된 정보 전송이다. 예를 들어, 인터넷의 인기 있는 응용은 그것의 WWW 페이지들의 브라우징이다. 그런 경우들에서, 통상적으로 검색수취된 정보는, HTTP에 따라 전송되는 바와 같이, HTML(HyperText Markup Language) 형식이다. 그러나, 다른 표준 마크업 언어들이 개발되고 있다. 그런 마크업 언어 중의 하나는 XML(eXtensible Markup Language)이다. XML은 XML 다큐먼트들로서 언급되는 데이터 객체들의 클래스를 설명하고, 그들을 처리하는 컴퓨터 프로그램들의 행위를 부분적으로 설명한다. HTML과 XML 사이의 제1 상이점은, 예를 들어, HTML 내에 정보 내용이 내용의 레이아웃과 짜여져 있어서 그들의 격리를 어렵게 만든다는 것이다. 반대로, XML 내에 저장 레이아웃과 내용의 논리 구조의 설명은 내용 자체로부터 분리되어 유지관리된다. 그러나, XML과 HTML 모두 SGML(Standard Generalized Markup Language)로서 알려진 마크업 언어의 변이들이다. 본 명세서에서 언급되는 바와 같이 XML은 일반적으로, 웹사이트 <http://www.w3.org>에 설명된 바와 같이, 임의의 XML 표준을 언급한다.

역행 호환성과 상호동작성을 유지하기 위해, 예를 들어, OPTIONS 응답에 확장 HTTP 헤더는 서버가 이진 디프들을 지원함을 클라이언트가 발견하도록 사용될 수 있다. GET 요구들에 확장 헤더는 클라이언트가 디프들을 수용함을 서버에게 통보한다.

클라이언트나 서버는 이진 디프를 사용하지 않기 위해 선택할 수 있다. 일부 경우들에서, 디프를 전송하는 메시지는(즉, HTTP 이진 디프 헤더) 그 파일 자체보다 더 클 수 있다. 그런 경우에, 서버와 클라이언트들 사이에 디프 대신에 다큐먼트를 전송하는 것은 더 바람직할 것이다. 클라이언트는 디프 크기가 새 파일보다 크다고 결정할 수 있다. 이것은, 예를 들어, 새 파일이 0 바이트를 가지면 생길 수 있다. 서버는 저장 공간을 절약하기 위해 디프들을 버리도록 결정할 수 있다. 디프가 사용되지 않으면, 전체 파일은 전송된다. 디프가 전송될 때 신호하기 위해, 클라이언트는 그것의 PUT 요구와 함께 확장 헤더(들)를 전송하여 몸체에 이진 디프의 존재 및 디프가 생성된 기본 파일의 버전 번호를 표시한다. 서버는 그것의 GET 응답과 함께 확장 헤더(들)를 전송하여 이진 디프 체인의 존재, 기본 파일의 버전 번호, 및 체인에 디프들의 수를 표시한다.

클라이언트 파일을 업데이트하기 위해 복수 디프들을 가져올 필요가 있을 때, 서버는 1 개의 답신에 디프들을 함께 체인하거나, 또는 디프들의 합이 새 버전 보다 크면 그 새 버전 자체를 전송하는 것 중에 하나를 선택할 수 있다.

디프들을 계산하고 적용하는 엔진은 바람직하게는 클라이언트들 및/또는 서버(들) 내에 제공된다. 발견 및 디프/버전 관리를 위한 프로토콜은 또한 바람직하게 구현된다. 확장 HTTP 헤더들은 상이점을 발견하는 것에 대해 클라이언트와 서버가 그들의 능력을 표현하도록 한다. 예를 들어, 클라이언트는 그것의 PUT 요구와 함께 확장 헤더(들)를 전송하여 몸체에 이진 디프의 존재 및 디프가 생성된 기본 파일의 버전 번호를 표시한다. 서버는 그것의 GET 응답과 함께 확장 헤더(들)를 전송하여 이진 디프 체인의 존재, 기본 파일의 버전 번호, 및 체인에 디프들의 수를 표시한다.

바람직하게는 서버 코드는 디프들을 적용하고(엔진을 사용함), 저장하고, 리턴하기 위해 디프들을 관리할 수 있고, 바람직하게는 각 클라이언트는 서버 상태를 유지관리하고, 디프들을 생성하고, 서버에 디프들을 전송하고, 리턴된 디프들을 적용하는 능력들을 갖는다.

복제는 다양한 응용 프로그램들에 의해 널리 사용되지만, 이런 시스템들의 비용과 성능은 계속 챌린지되고 있다. 본 발명은 복제의 특성(알려진 클라이언트/서버 상태)을 사용하여 시스템의 효율성에서 주된 도약을 한다. 본 발명은 서버 기반 다큐먼트들이나 파일들이 오프라인으로 액세스되도록 하는 제품들과 같은 반복적으로 업데이트되는 큰 파일들을 복제하는 시스템들에 적용될 수 있다.

상술된 바와 같이, 본 발명의 실시예들은 다양한 컴퓨팅 디바이스들과 연결되어 설명되는 한편, 기반되는 개념들은 임의의 컴퓨팅 디바이스나 시스템에 적용될 수 있다.

본 명세서에 설명된 다양한 기술들은 하드웨어나 소프트웨어 또는, 적절하면, 그 둘의 조합에 연결되어 구현될 수 있다. 그러므로, 본 발명의 방법들과 장치 또는 그것들의 특정 양태들이나 부분들은, 플로피 디스켓들, CD-ROM들, 하드 드라이브들, 또는 임의의 다른 머신 판독가능 저장 매체와 같은 실체의 매체들에 구현된 프로그램 코드(즉, 명령들)의 형태를 가질

것이고, 프로그램 코드가 컴퓨터와 같은 머신에 로딩되고 실행될 때 그 머신은 본 발명을 실시하는 장치가 된다. 프로그램 가능한 컴퓨터들에의 프로그램 코드 실행의 경우, 컴퓨팅 디바이스는 일반적으로 처리기, 처리기에 의해 판독가능한 저장 매체(휘발성과 비휘발성 메모리 및/또는 저장 소자들을 포함함), 적어도 1 개의 입력 디바이스, 및 적어도 1 개의 출력 디바이스를 포함한다. 프로그램(들)은 원하면 어셈블리어 또는 기계어로 구현될 수 있다. 임의의 경우에, 언어는 컴파일 또는 인터프리트되는 언어일 것이고, 하드웨어 구현들과 조합될 수 있다.

본 발명의 방법들과 장치는, 광섬유나 임의의 다른 형태의 전송을 통해 전기 도선이나 케이블링과 같은 특정 전송 매체를 통해 전송되는 프로그램 코드의 형태로 구현되는 통신을 통해 또한 실시될 수 있고, 프로그램 코드가 EPROM, 게이트 배열, 프로그램가능한 논리 디바이스(PLD), 클라이언트 컴퓨터 등과 같은 머신에 의해 수신되고, 로딩되고, 실행될 때, 머신은 본 발명을 실시하는 장치가 된다. 일반 목적 처리기에 구현될 때, 프로그램 코드는 처리기와 조합하여 본 발명의 기능을 호출하기 위해 동작하는 특정 장치를 제공한다. 추가로, 본 발명과 연결해서 사용된 임의의 저장 기술들은 모두 하드웨어와 소프트웨어의 조합일 것이다.

본 발명이 다양한 도면들의 바람직한 실시예들과 연결하여 설명되는 한편, 다른 유사 실시예들이 사용될 수 있거나, 수정본들과 추가본들이 본 발명에서 벗어나지 않고 본 발명의 동일 기능을 수행하는 설명된 실시예들에 만들어질 수 있음을 이해할 수 있다. 그러므로, 본 발명은 임의의 1 개의 실시예에 제한되어서는 안되고, 그대신, 첨부된 청구 범위에 따라 범위가 이해되어야 한다.

발명의 효과

본 발명은 파일에 변경이 가해지면 클라이언트와 서버 기반의 파일의 복사본들을 동기화시켜서 유지관리하는 메카니즘을 제공한다. 데이터는 클라이언트와 서버에게 모두 알려진 이전 버전과 비교되고, 그 둘 사이에 상이점들의 고압축 표현이 생성된다. 이런 상이점들 또는 디프들은 전송되고 HTTP 프로토콜 확장을 사용할 수 있다.

(57) 청구의 범위

청구항 1.

업데이트된 파일을 유지관리하는 방법에 있어서,

제1 디바이스에서 기본 파일의 제1 복사본과 제2 복사본을 저장하는 단계;

상기 제1 디바이스에 상기 제1 복사본에의 변경들을 수신하는 단계;

상기 변경된 제1 복사본과 상기 제2 복사본 사이의 제1 상이점을 결정하는 단계;

서버에 상기 제1 상이점을 전송하는 단계; 및

상기 서버에 있는 기본 파일이 상기 제1 디바이스에 저장된 상기 기본 파일과 동일하면, 상기 서버에서 상기 제1 상이점을 수용하고, 그렇지 않으면 상기 서버에서 상기 제1 상이점을 거절하는 단계

를 포함하는 방법.

청구항 2.

제1항에 있어서, 상기 제1 디바이스에 상기 파일의 상기 제1 복사본과 상기 제2 복사본을 저장하기 전에, 상기 서버로부터 상기 제1 디바이스에서 상기 파일의 상기 제1 복사본을 수신하는 단계, 및 상기 제1 디바이스에서 상기 파일의 제2 복사본을 만드는 단계를 더 포함하는 방법.

청구항 3.

제1항에 있어서, 상기 제1 상이점을 결정하는 단계는 상기 변경된 제1 복사본과 상기 제2 복사본 사이의 이진 비교들을 사용하는 단계를 포함하는 방법.

청구항 4.

제1항에 있어서, 상기 제1 상이점은 이진 디프(binary diff)인 방법.

청구항 5.

제1항에 있어서, 상기 상이점이 상기 서버에서 거절되면, 상기 서버로부터 상기 제1 디바이스로 제2 상이점을 전송하고, 상기 제1 디바이스에 저장된 상기 기본 파일의 상기 제2 복사본에 상기 제2 상이점을 적용하는 방법.

청구항 6.

제1항에 있어서, 상기 서버에 있는 기본 파일이 상기 제1 디바이스에 저장된 상기 기본 파일과 동일한지를 판정하는 단계를 더 포함하는 방법.

청구항 7.

제1항에 있어서, 상기 제1 상이점을 전송하는 단계는 HTTP 프로토콜의 확장들을 사용하는 단계를 포함하는 방법.

청구항 8.

제1항에 있어서,

제2 디바이스에 상기 기본 파일의 제1 복사본과 제2 복사본을 저장하는 단계;

상기 제2 디바이스에서 상기 제1 복사본에의 변경들을 수신하는 단계;

상기 변경된 제1 복사본과 상기 제2 복사본 사이의 제2 상이점을 결정하는 단계;

상기 제2 상이점을 상기 서버에 전송하는 단계; 및

상기 서버에 있는 기본 파일이 상기 제2 디바이스에 저장된 상기 기본 파일과 동일하면, 상기 서버에서 상기 제2 상이점을 수용하고, 그렇지 않으면 상기 서버에서 상기 제2 상이점을 거절하는 단계

를 더 포함하는 방법.

청구항 9.

제8항에 있어서, 상기 서버에서 상기 제2 상이점을 거절하면, 제3 상이점을 상기 제2 디바이스에 전송하고, 상기 제2 디바이스에 저장된 상기 기본 파일의 상기 제2 복사본에 상기 제3 상이점을 적용하는 단계를 더 포함하는 방법.

청구항 10.

제8항에 있어서, 상기 제2 디바이스에 상기 파일의 상기 제1 복사본과 상기 제2 복사본을 저장하기 전에, 상기 서버로부터 상기 제2 디바이스에서 상기 파일의 상기 제1 복사본을 수신하고, 상기 제2 디바이스에서 상기 파일의 제2 복사본을 만드는 단계를 더 포함하는 방법.

청구항 11.

제8항에 있어서, 상기 제2 상이점을 결정하는 단계는 상기 제2 디바이스에서 상기 변경된 제1 복사본과 상기 제2 복사본 사이의 이진 비교들을 사용하는 단계를 포함하는 방법.

청구항 12.

제8항에 있어서, 상기 제2 상이점은 이진 디프인 방법.

청구항 13.

제8항에 있어서, 상기 서버에 있는 기본 파일이 상기 제2 디바이스에 저장된 상기 기본 파일과 동일한지를 결정하는 단계를 더 포함하는 방법.

청구항 14.

제8항에 있어서, 상기 제2 상이점을 전송하는 단계는 HTTP 프로토콜의 확장들을 사용하는 단계를 포함하는 방법.

청구항 15.

제1 디바이스에 기본 파일의 제1 복사본과 제2 복사본을 저장하는 단계;

상기 제1 디바이스에서 상기 제1 복사본에의 변경들을 수신하는 단계;

상기 변경된 제1 복사본과 상기 제2 복사본 사이의 제1 상이점을 결정하는 단계;

상기 제1 상이점을 서버에 전송하는 단계; 및

상기 서버에 있는 상기 기본 파일이 상기 제1 디바이스에 저장된 상기 기본 파일과 동일하면, 상기 서버에서 상기 제1 상이점을 수용하고, 그렇지 않으면 상기 서버에서 상기 제1 상이점을 거절하는 단계

를 포함하는 업데이트된 파일을 유지관리하는 방법을 수행하는 컴퓨터 실행가능 명령들이 저장되어 있는 컴퓨터 판독가능 매체.

청구항 16.

제15항에 있어서, 상기 제1 디바이스에 상기 파일의 상기 제1 복사본과 상기 제2 복사본을 저장하기 전에, 상기 서버로부터 상기 제1 디바이스에서 상기 파일의 상기 제1 복사본을 수신하고, 상기 제1 디바이스에서 상기 파일의 제2 복사본을 만들기 위한 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 17.

제15항에 있어서, 상기 제1 상이점을 결정하는 단계는 상기 변경된 제1 복사본과 상기 제2 복사본 사이의 이진 비교들을 사용하는 단계를 포함하는 컴퓨터 판독 가능 매체.

청구항 18.

제15항에 있어서, 상기 제1 상이점은 이진 디프인 컴퓨터 판독 가능 매체.

청구항 19.

제15항에 있어서, 상기 상이점이 상기 서버에서 거절되면, 상기 서버로부터 상기 제1 디바이스에 제2 상이점을 전송하고, 상기 제1 디바이스에 저장된 상기 기본 파일의 상기 제2 복사본에 상기 제2 상이점을 적용하는 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 20.

제15항에 있어서, 상기 서버에 있는 기본 파일이 상기 제1 디바이스에 저장된 상기 기본 파일과 동일한지를 결정하는 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 21.

제15항에 있어서, 상기 제1 상이점을 전송하는 단계는 HTTP 프로토콜의 확장들을 사용하는 단계를 포함하는 컴퓨터 판독가능 매체.

청구항 22.

제15항에 있어서,

제2 디바이스에 상기 기본 파일의 제1 복사본과 제2 복사본을 저장하는 단계;

상기 제2 디바이스에서 상기 제1 복사본에의 변경들을 변경들을 수신하는 단계;

상기 변경된 제1 복사본과 상기 제2 복사본 사이의 제2 상이점을 결정하는 단계;

상기 제2 상이점을 상기 서버에 전송하는 단계; 및

상기 서버에 있는 기본 파일이 상기 제2 디바이스에 저장된 상기 기본 파일과 동일하면, 상기 서버에서 상기 제2 상이점을 수용하고, 그렇지 않으면 상기 서버에서 상기 제2 상이점을 거절하는 단계

를 위한 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 23.

제22항에 있어서, 상기 제2 상이점이 상기 서버에서 거절되면, 상기 제2 디바이스에 제3 상이점을 전송하고, 상기 제2 디바이스에 저장된 상기 기본 파일의 제2 복사본에 상기 제3 상이점을 적용하는 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 24.

제22항에 있어서, 상기 제2 디바이스에 상기 파일의 상기 제1 복사본과 상기 제2 복사본을 저장하기 전에, 상기 서버로부터 상기 제2 디바이스에서 상기 파일의 상기 제1 복사본을 수신하고, 상기 제2 디바이스에서 상기 파일의 제2 복사본을 만드는 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 25.

제22항에 있어서, 상기 제2 상이점을 결정하는 단계는 상기 제2 디바이스에서 상기 변경된 제1 복사본과 상기 제2 복사본 사이의 이진 비교들을 사용하는 단계를 포함하는 컴퓨터 판독가능 매체.

청구항 26.

제22항에 있어서, 상기 제2 상이점은 이진 디프인 컴퓨터 판독가능 매체.

청구항 27.

제22항에 있어서, 상기 서버에 있는 기본 파일이 상기 제2 디바이스에 저장된 상기 기본 파일과 동일한지를 판정하는 컴퓨터 실행가능 명령들을 더 포함하는 컴퓨터 판독가능 매체.

청구항 28.

제22항에 있어서, 상기 제2 상이점을 전송하는 단계는 HTTP 프로토콜의 확장들을 사용하는 단계를 포함하는 컴퓨터 판독가능 매체.

청구항 29.

파일 동기화 시스템에 사용을 위한 시스템에 있어서,

기본 파일의 제1 복사본과 제2 복사본을 저장하는 저장 디바이스;

상기 기본 파일의 상기 제1 복사본에의 변경들을 수신하는 입력 디바이스; 및

상기 수신된 변경들에 기초한 상기 제1 복사본을 변경하고, 상기 변경된 제1 복사본과 상기 제2 복사본 사이의 상이점을 결정하는 처리기

를 포함하는 시스템.

청구항 30.

제29항에 있어서, 상기 상이점은 이진 상이점인 시스템.

청구항 31.

제29항에 있어서, 상기 기본 파일은 서버에 상주하고, 상기 서버에 상기 상이점을 전송하기 위한 출력 디바이스를 더 포함하는 시스템.

청구항 32.

제29항에 있어서, 상기 상이점은 HTTP 프로토콜의 확장들을 사용하여 상기 서버에 전송되는 시스템.

청구항 33.

제29항에 있어서, 제2 상이점을 수신하는 디바이스를 더 포함하고, 상기 처리기는 상기 기본 파일의 상기 제2 복사본에 상기 제2 상이점을 적용하여 수정된 제2 복사본을 생성하고, 상기 수정된 제2 복사본에 상기 수신된 변경들을 적용하여 제3 복사본을 생성하고, 그리고 상기 제3 복사본과 상기 수정된 제2 복사본 사이의 제3 상이점을 결정하는 시스템.

청구항 34.

제33항에 있어서, 서버에 상기 제3 상이점을 전송하는 출력 디바이스를 더 포함하는 시스템.

청구항 35.

파일 동기화 시스템에 사용을 위한 시스템에 있어서,

기본 파일을 저장하는 저장 디바이스;

클라이언트 디바이스로부터 상이점을 수신하는 입력 디바이스; 및

상기 상이점이 상기 저장된 기본 파일에 적용가능한지를 판정하고, 그렇다면 상기 상이점을 상기 기본 파일에 적용하고, 그렇지 않으면 상기 클라이언트 디바이스에게 상기 상이점이 거절되었음을 지시하는 처리기

를 포함하는 시스템.

청구항 36.

제35항에 있어서, 상기 상이점은 이진 상이점인 시스템.

청구항 37.

제35항에 있어서, 상기 상이점은 HTTP 프로토콜의 확장들의 형태로 수신되는 시스템.

청구항 38.

제35항에 있어서, 상기 상이점이 상기 처리기에 의해 거절되면, 제2 상이점이 상기 저장 디바이스로부터 검색수취되고 출력 디바이스를 통해 상기 클라이언트 디바이스에 전송되는 시스템.

청구항 39.

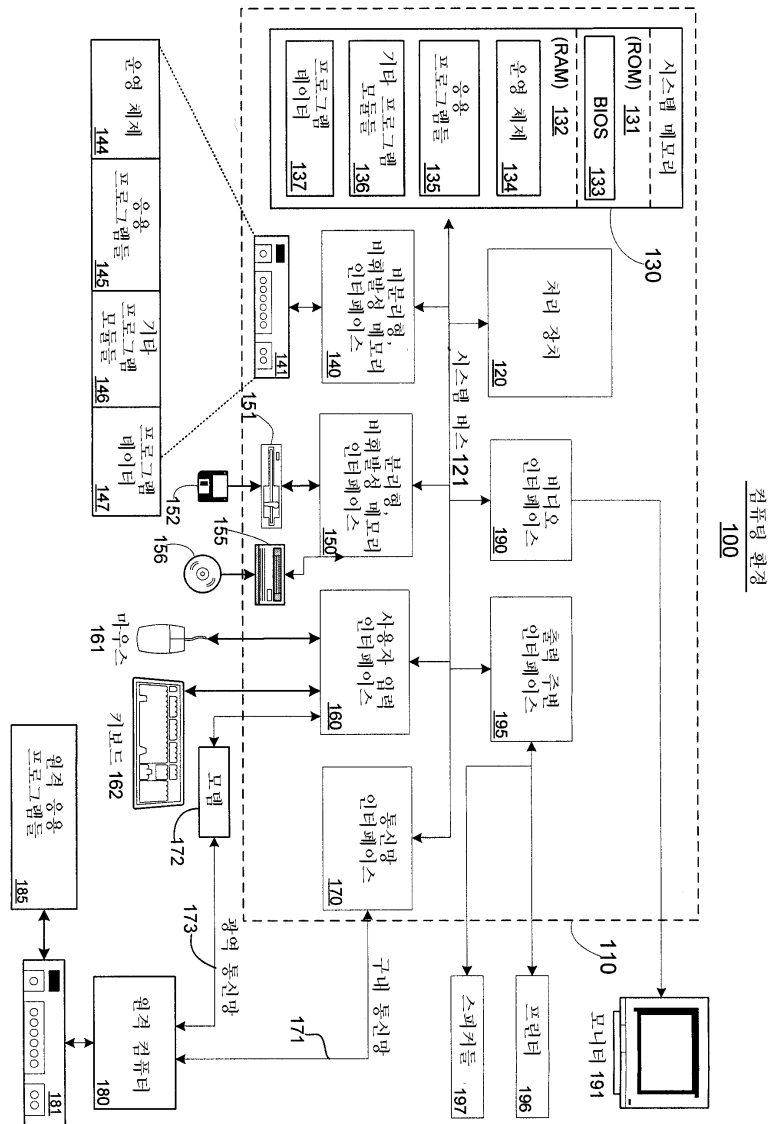
제38항에 있어서, 상기 제2 상이점은 제2 클라이언트 디바이스로부터 수신되는 시스템.

청구항 40.

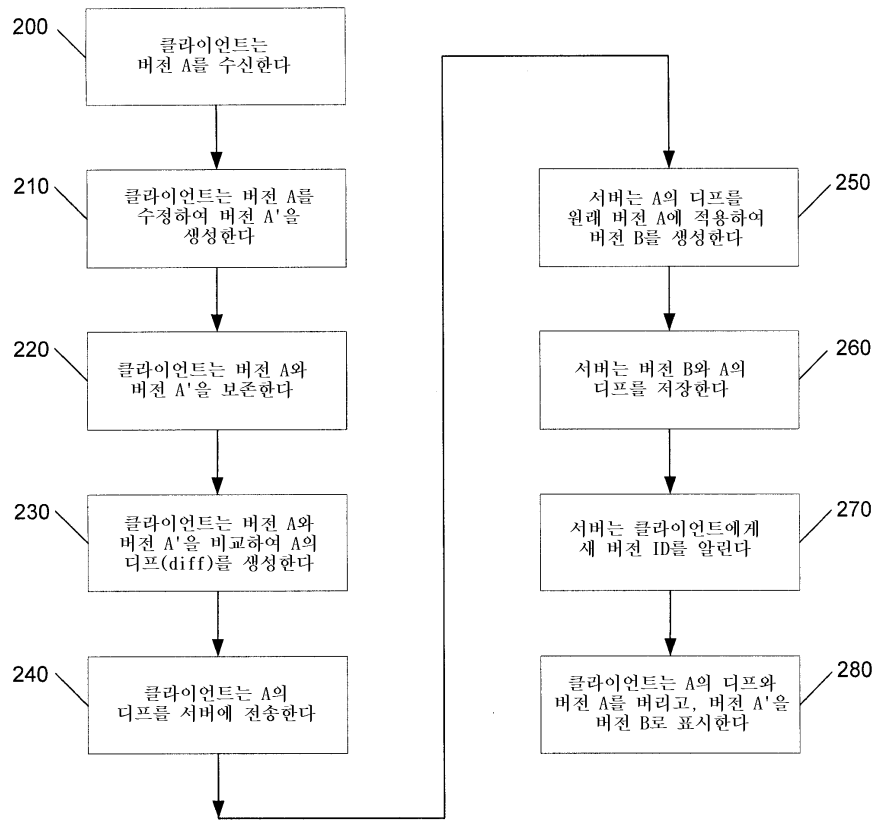
제35항에 있어서, 상기 클라이언트 디바이스가 상기 상이점을 생성하기 위해 사용한 클라이언트 기본 파일과 상기 기본 파일이 동일하면, 상기 상이점이 상기 저장된 기본 파일에 적용가능한 시스템.

도면

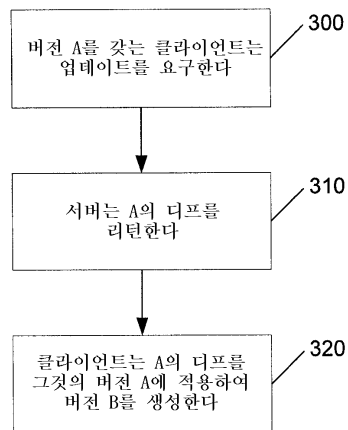
도면1



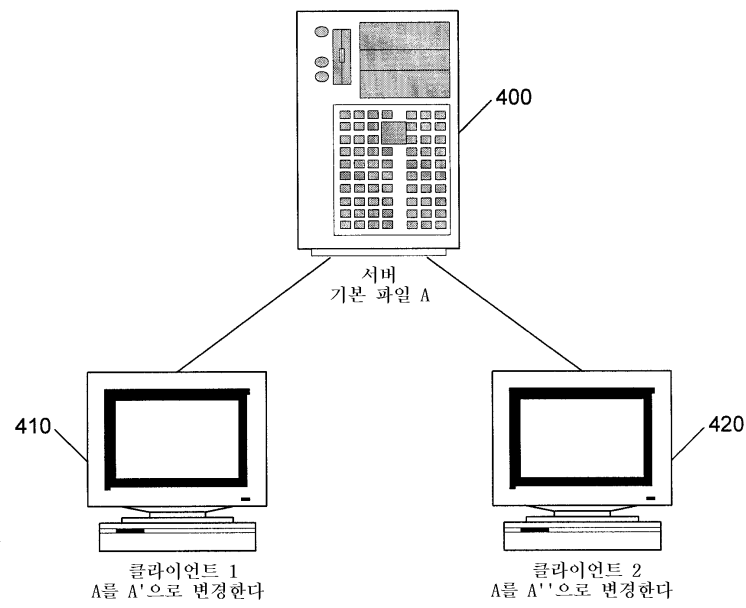
도면2



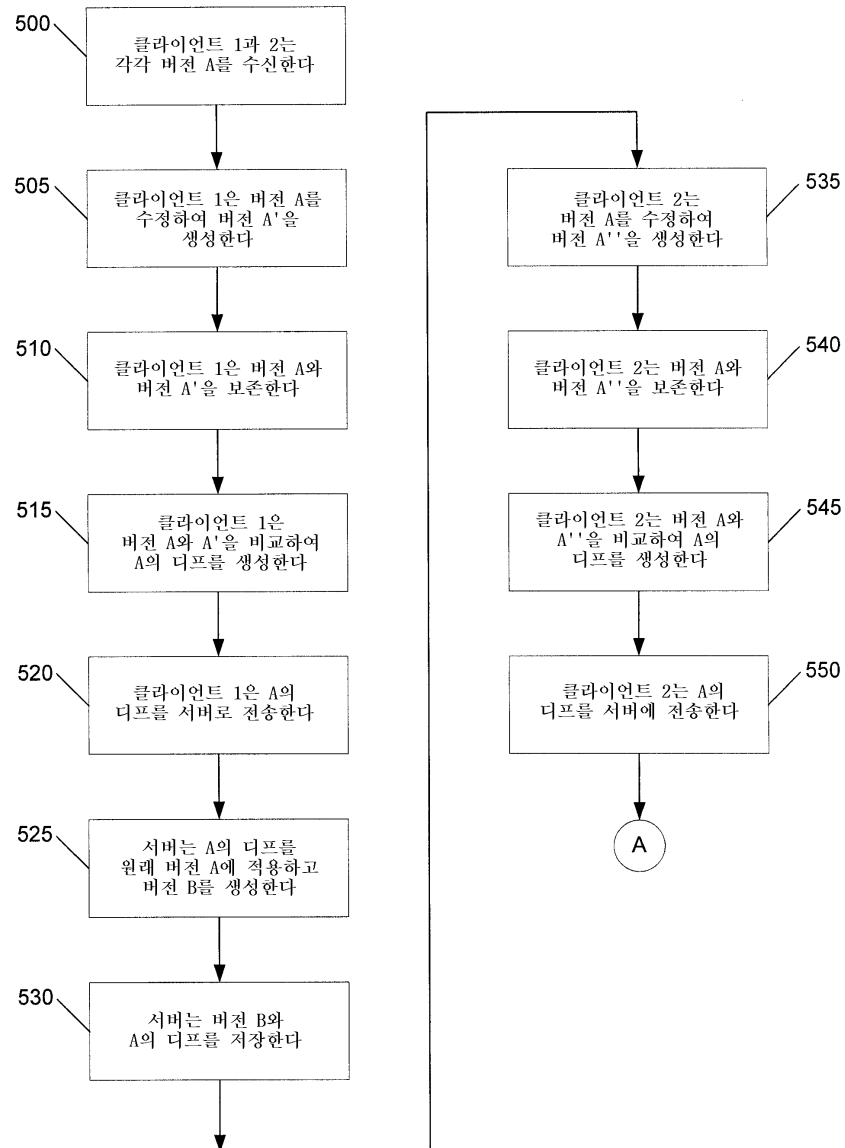
도면3



도면4



도면5



도면6

