(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2020-0093561
(43) 공개일자 2020년08월05일

(51) 국제특허분류(Int. Cl.)
G06F 16/178 (2019.01) *G06F 16/11* (2019.01)
G06F 16/17 (2019.01) *G06F 16/176* (2019.01)
(52) CPC특허분류
G06F 16/178 (2019.01)
G06F 16/11 (2019.01)
(21) 출원번호 10-2020-7016022
(22) 출원일자(국제) 2018년12월10일
심사청구일자 2020년06월04일
(85) 번역문제출일자 2020년06월04일
(86) 국제출원번호 PCT/US2018/064670
(87) 국제공개번호 WO 2019/133229
국제공개일자 2019년07월04일
(30) 우선권주장
62/611,473 2017년12월28일 미국(US)
(뒷면에 계속)

(71) 출원인
드롭박스, 인크.
미국 94158 캘리포니아주 샌프란시스코 스위트 200 오웬스 스트리트 1800
(72) 발명자
잉, 로버트
미국 94107 캘리포니아주 샌프란시스코 브라난 스트리트 333 드롭박스 인크.
쿠라파티, 니푼
미국 94107 캘리포니아주 샌프란시스코 브라난 스트리트 333 드롭박스 인크.
굽타, 가우탐
미국 94107 캘리포니아주 샌프란시스코 브라난 스트리트 333 드롭박스 인크.
(74) 대리인
양영준, 김연송, 백만기

전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **클라이언트 동기화 서비스를 위한 원격 트리 업데이트**

(57) 요 약

개시된 기술은 콘텐츠 관리 시스템으로부터 동작 로그를 포함하는 동작 데이터를 수신하고, 상기 동작 로그를 실행하며, 상기 동작 로그의 실행에 기초하여 상기 콘텐츠 관리 시스템에 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리를 업데이트하도록 구성된 시스템에 관한 것이다.

(52) CPC특허분류
G06F 16/1734 (2019.01)
G06F 16/1767 (2019.01)

(30) 우선권주장
15/863,751 2018년01월05일 미국(US)
15/863,748 2018년01월05일 미국(US)

명 세 서

청구범위

청구항 1

컴퓨터로 구현되는 방법으로서,

콘텐츠 관리 시스템으로부터 동작 데이터를 수신하는 단계로서, 상기 동작 데이터는 동작 로그(log)를 포함하는, 상기 동작 데이터를 수신하는 단계;

상기 동작 로그를 실행하는 단계; 및

상기 동작 로그의 실행에 기초하여, 상기 콘텐츠 관리 시스템에 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리를 업데이트하는 단계를 포함하는, 컴퓨터로 구현되는 방법.

청구항 2

제1항에 있어서, 상기 콘텐츠 관리 시스템에 커서를 전송하는 단계를 더 포함하고, 상기 동작 로그는 상기 커서에 기초하는, 컴퓨터로 구현되는 방법.

청구항 3

제1항에 있어서, 상기 동작 데이터는 다수의 네임스페이스에 걸쳐 선형화된 동작을 포함하는, 컴퓨터로 구현되는 방법.

청구항 4

제1항에 있어서, 상기 동작 데이터는 네임스페이스에 대한 리비전(revision) 데이터를 포함하는, 컴퓨터로 구현되는 방법.

청구항 5

제1항에 있어서,

상기 서버 상태와 파일 시스템 상태 사이의 알려진 동기화된 상태를 나타내는 동기 트리와 상기 원격 트리 사이의 차이에 기초하여 상기 서버 상태와 상기 파일 시스템 상태가 동기화 상태에 있지 않은 것으로 결정하는 단계;

상기 차이에 기초하여, 상기 서버 상태와 상기 파일 시스템 상태를 수렴하도록 구성된 동작 세트를 생성하는 단계; 및

상기 동작 세트의 실행을 관리하는 단계를 더 포함하는, 컴퓨터로 구현되는 방법.

청구항 6

제5항에 있어서, 상기 원격 트리와 상기 동기 트리는 클라이언트 디바이스에 저장되는, 컴퓨터로 구현되는 방법.

청구항 7

제1항에 있어서,

상기 콘텐츠 관리 시스템으로부터, 원격 트리에 표현된 현존하는 네임스페이스 내 타깃 네임스페이스를 마운트(mount)하기 위한 마운트 통지를 수신하는 단계;

상기 콘텐츠 관리 시스템으로부터, 상기 타깃 네임스페이스에 대한 동작 로그의 초기 부분을 수신하는 단계;

상기 동작 로그의 초기 부분에 기초하여 상기 타깃 네임스페이스에 대한 서브 트리를 구축하는 단계; 및

상기 원격 트리의 마운트 위치에서 상기 타깃 네임스페이스에 대한 서브 트리를 마운트하는 단계를 더 포함하는, 컴퓨터로 구현되는 방법.

청구항 8

제7항에 있어서, 상기 동작 로그의 초기 부분은 상기 동작 로그의 시작에 의해 상기 콘텐츠 관리 시스템에 의해 검출된 마운트 동작을 위한 커서 값으로 한정되는, 컴퓨터로 구현되는 방법.

청구항 9

제7항에 있어서, 상기 타깃 네임스페이스에 대한 서브 트리는 상기 동작 로그의 초기 부분이 상기 서브 트리를 구축하도록 처리된 후에 마운트되는, 컴퓨터로 구현되는 방법.

청구항 10

제7항에 있어서,

상기 콘텐츠 관리 시스템으로부터, 상기 타깃 네임스페이스에 대한 마운트 동작을 수신하는 단계를 더 포함하고, 상기 마운트 동작은 상기 현존하는 네임스페이스와 연관되고 상기 현존하는 네임스페이스에서 상기 마운트 위치를 지정하고;

상기 타깃 네임스페이스는 상기 마운트 동작에 응답하여 상기 마운트 위치에서 마운트되는, 컴퓨터로 구현되는 방법.

청구항 11

명령을 포함하는 비-일시적인 컴퓨터 판독 가능 매체로서, 상기 명령은 컴퓨팅 시스템에 의해 실행될 때 상기 컴퓨팅 시스템으로 하여금,

콘텐츠 관리 시스템으로부터 동작 데이터를 수신하게 하되, 상기 동작 데이터는 동작 로그를 포함하고;

상기 동작 로그를 실행하게 하고;

상기 동작 로그의 실행에 기초하여, 상기 콘텐츠 관리 시스템에 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리를 업데이트하게 하는, 비-일시적인 컴퓨터 판독 가능 매체.

청구항 12

제11항에 있어서, 상기 명령은 상기 컴퓨팅 시스템으로 하여금 커서를 상기 콘텐츠 관리 시스템으로 전송하게 하고, 상기 동작 로그는 상기 커서에 기초하는, 비-일시적인 컴퓨터 판독 가능 매체.

청구항 13

제11항에 있어서, 상기 동작 데이터는 다수의 네임스페이스에 걸쳐 선형화된 동작을 포함하는, 비-일시적인 컴퓨터 판독 가능 매체.

청구항 14

제11항에 있어서, 상기 동작 데이터는 네임스페이스에 대한 리비전 데이터를 포함하는, 비-일시적인 컴퓨터 판독 가능 매체.

청구항 15

제11항에 있어서, 상기 명령은 상기 컴퓨팅 시스템으로 하여금,

상기 서버 상태와 파일 시스템 상태 사이의 알려진 동기화된 상태를 나타내는 동기 트리와 상기 원격 트리 사이의 차이에 기초하여 상기 서버 상태와 상기 파일 시스템 상태가 동기화된 상태에 있지 않은 것으로 결정하게 하고;

상기 차이에 기초하여, 상기 서버 상태와 상기 파일 시스템 상태를 수렴하도록 구성된 동작 세트를 생성하게 하고;

상기 동작 세트의 실행을 관리하게 하는, 비-일시적인 컴퓨터 판독 가능 매체.

청구항 16

시스템으로서,

프로세서; 및

명령을 저장하는 비-일시적인 컴퓨터 판독 가능 매체를 포함하고, 상기 명령은, 상기 프로세서에 의해 실행될 때, 상기 프로세서로 하여금,

콘텐츠 관리 시스템으로부터 동작 데이터를 수신하게 하고, 상기 동작 데이터는 동작 로그를 포함하고;

상기 동작 로그를 실행하게 하고;

상기 동작 로그의 실행에 기초하여, 상기 콘텐츠 관리 시스템에 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리를 업데이트하게 하는, 시스템.

청구항 17

제16항에 있어서, 상기 명령은 상기 프로세서로 하여금 커서를 상기 콘텐츠 관리 시스템으로 전송하게 하고, 상기 동작 로그는 상기 커서에 기초하는, 시스템.

청구항 18

제16항에 있어서, 상기 동작 데이터는 다수의 네임스페이스에 걸쳐 선형화된 동작을 포함하는, 시스템.

청구항 19

제16항에 있어서, 상기 동작 데이터는 네임스페이스에 대한 리비전 데이터를 포함하는, 시스템.

청구항 20

제16항에 있어서, 상기 명령은 상기 프로세서로 하여금,

상기 서버 상태와 파일 시스템 상태 사이의 알려진 동기화된 상태를 나타내는 동기 트리와 상기 원격 트리 사이의 차이에 기초하여 상기 서버 상태와 상기 파일 시스템 상태가 동기화된 상태에 있지 않은 것으로 결정하게 하고;

상기 차이에 기초하여, 상기 서버 상태와 상기 파일 시스템 상태를 수렴하도록 구성된 동작 세트를 생성하게 하고;

상기 동작 세트의 실행을 관리하게 하는, 시스템.

발명의 설명

기 술 분 야

[0001] **관련 출원에 대한 상호 참조**

[0002] 본 출원은 2018년 1월 5일자로 출원된 미국 가특허 출원 번호 15/863,751, 2018년 1월 5일자로 출원된 미국 가특허 출원 번호 15/863,748 및 2017년 12월 28일자로 출원된 미국 가출원 번호 62/611,473의 우선권을 주장하며, 이들 선출원 문헌 모두는 전체 내용이 본 명세서에 병합된다.

배 경 기 술

[0003] 콘텐츠 관리 시스템은 사용자가 네트워크를 사용하여 다수의 디바이스에 걸쳐 콘텐츠 아이템(content item)에 액세스하고 관리할 수 있게 한다. 일부 콘텐츠 관리 시스템은 사용자가 콘텐츠 아이템을 공유하고 사용자가 콘텐츠 아이템을 사용하여 협업하는 데 도움이 되는 추가 기능을 제공하게 할 수 있다. 콘텐츠 관리 시스템은 일반적으로 콘텐츠 아이템을 서버에 저장하고 사용자가 네트워크를 통해 콘텐츠 아이템에 액세스할 수 있게 한다. 일부 콘텐츠 관리 시스템은 또한 사용자에게 보다 자연스러운 인터페이스(예를 들어, 기본 애플리케이션 또는 클라이언트 디바이스의 파일 시스템 내 인터페이스)에서 콘텐츠 아이템에 대한 빠른 액세스를 제공하기 위해 로컬 사본을 클라이언트 디바이스에 저장하게 할 수 있다. 추가적으로, 이것은 사용자가 오프라인일 때 사용자가 콘텐츠 아이템에 액세스할 수 있게 한다. 콘텐츠 관리 시스템은 다수의 클라이언트 디바이스와 서버에 걸쳐 콘텐츠 아이템의 사본을 동기화하여 각 사본이 동일하도록 한다. 그러나, 콘텐츠 아이템의 동기화는 어렵고 다수

의 기술적 장애물과 연관된다.

도면의 간단한 설명

[0004] 본 기술의 상기 언급된 및 다른 장점 및 특징은 첨부 도면에 도시된 특정 구현을 참조하여 명백해질 것이다. 이 기술 분야에 통상의 지식을 가진 자라면 이들 도면이 본 기술의 일부 예만을 나타내는 것일 뿐 본 기술의 범위를 이들 예로 제한하는 것이 아니라는 것을 이해할 수 있을 것이다. 또한, 이 기술 분야에 통상의 지식을 가진 자라면 첨부 도면을 사용하여 추가적인 특성 및 상세와 함께 기술되고 설명된 바와 같은 본 기술의 원리를 이해할 수 있을 것이다.

도 1은 일부 실시형태에 따른 콘텐츠 관리 시스템과 클라이언트 디바이스의 일례를 도시한 도면;

도 2는 일부 실시형태에 따른 클라이언트 동기화 서비스의 일례를 도시한 도면;

도 3은 다양한 실시형태에 따른 트리 데이터 구조(tree data structure)의 일례를 도시한 도면;

도 4는 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한 도면;

도 5는 본 기술의 다양한 실시형태에 따라 트리 데이터 구조를 사용하여 서버 상태와 파일 시스템 상태를 동기화하는 예시적인 방법을 도시한 도면;

도 6은 본 기술의 다양한 실시형태에 따라 트리 데이터 구조를 사용하여 서버 상태와 파일 시스템 상태를 동기화할 때 충돌을 해결하기 위한 예시적인 방법을 도시한 도면;

도 7은 다양한 실시형태에 따라 추가 동작(add operation)에 대한 규칙 위반을 나타내는 트리 데이터 구조의 일례를 도시한 도면;

도 8은 본 기술의 다양한 실시형태에 따라 서버 상태와 파일 시스템 상태를 증분적으로 수렴하기 위한 예시적인 방법을 도시한 도면;

도 9는 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한 도면;

도 10은 예시적인 시나리오를 도시한 도면;

도 11은 본 기술의 다양한 실시형태에 따른 2개의 동작 계획의 예시적인 벤 다이어그램(Venn diagram) 표현을 도시한 도면;

도 12는 본 기술의 다양한 실시형태에 따른, 동작 계획의 변경을 관리하기 위한 예시적인 방법을 도시한 도면;

도 13은 본 기술의 다양한 실시형태에 따른 예시적인 시나리오를 도시한 도면;

도 14는 본 기술의 다양한 실시형태에 따라 로컬 트리(local tree)를 업데이트하기 위한 예시적인 방법을 도시한 도면;

도 15는 본 기술의 다양한 실시형태에 따른 이동(move) 또는 이름 변경(rename) 동작에 응답하여 로컬 트리를 업데이트하는 예시적인 방법을 도시한 도면;

도 16은 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한 도면;

도 17은 다양한 실시형태에 따른 네임스페이스(namespace)를 마운트(mount)하는 개념도를 도시한 도면;

도 18은 본 기술의 다양한 실시형태에 따라 원격 트리(remote tree)에서 네임스페이스를 마운트하기 위한 예시적인 방법을 도시한 도면;

도 19a는 본 기술의 다양한 실시형태에 따라, 콘텐츠 관리 시스템과 클라이언트 디바이스 사이에서 콘텐츠를 동기화하기 위한 예시적인 아키텍처를 도시한 개략도;

도 19b는 본 기술의 다양한 실시형태에 따라, 콘텐츠 관리 시스템과 클라이언트 디바이스 사이에서 콘텐츠를 동기화하기 위한 예시적인 아키텍처에서 콘텐츠 아이템의 블록을 저장하고 추적하기 위한 예시적인 구성을 도시한 도면;

도 19c는 본 기술의 다양한 실시형태에 따른, 콘텐츠 관리 시스템 상의 서버 파일 저널(server file journal)과 클라이언트 디바이스 사이의 파일 저널 인터페이스에 의해 처리되는 예시적인 통신을 도시한 도면;

도 19d는 본 기술의 다양한 실시형태에 따라, 콘텐츠 관리 시스템 상의 서버 파일 저널과 클라이언트 디바이스 사이의 통신을 변환하기 위한 예시적인 프로세스를 도시한 도면;

도 20a는 본 기술의 다양한 실시형태에 따라, 서버 파일 저널 데이터를 선형화된 동작으로 변환하기 위한 예시적인 변환 및 선형화 프로세스를 도시한 도면;

도 20b는 본 기술의 다양한 실시형태에 따라, 클라이언트 디바이스로부터 서버 파일 저널에 대한 리비전(revision)으로 동작을 변환하기 위한 예시적인 변환 및 선형화 프로세스를 도시한 도면;

도 20c는 콘텐츠 관리 시스템 상의 서버 파일 저널로부터 클라이언트 디바이스에 대한 동작으로 리비전을 변환하기 위한 예시적인 방법을 도시한 도면;

도 21은 본 기술의 다양한 실시형태에 따른 네임스페이스 간 동작(cross-namespace operation)의 예시적인 선형화를 도시한 도면;

도 22는 본 기술의 다양한 실시형태에 따른, 이벤트에 대해 계산된 램포트 클록(Lamport clock)에 따라 정렬된 네임스페이스에 걸친 이벤트를 도시한 도면; 및

도 23은 본 기술의 특정 양태를 구현하기 위한 시스템의 일례를 도시한 도면.

발명을 실시하기 위한 구체적인 내용

[0005] 본 기술의 다양한 예가 이하에서 상세히 논의된다. 특정 구현들이 논의되지만, 이것은 예시의 목적으로만 이루어진 것이라는 것을 이해해야 한다. 이 기술 분야에서 통상의 지식을 가진 자라면 본 기술의 사상 및 범위를 벗어나지 않고 다른 구성 요소 및 구성이 사용될 수 있다는 것을 인식할 수 있을 것이다.

[0006] 컴퓨팅 및 네트워킹 기술의 다양한 진보는 콘텐츠 관리 시스템이 사용자에게 다수의 디바이스에 걸쳐 콘텐츠 아이템에 대한 액세스를 제공할 수 있게 할 수 있다. 콘텐츠 아이템은 파일, 문서, 메시지(예를 들어, 이메일 메시지 또는 문자 메시지), 다수의 파일을 포함하는 미디어 파일(예를 들어, 사진, 비디오 및 오디오 파일) 폴더, 또는 임의의 다른 콘텐츠 유닛을 포함할 수 있지만 이로 제한되는 것은 아니다. 콘텐츠 아이템은 다수의 사용자와 공유되거나 편집, 삭제, 추가, 이름 변경 또는 이동될 수 있다. 그러나 여러 컴퓨팅 디바이스(예를 들어, 서버 및 클라이언트 디바이스)에 걸쳐 그리고 여러 사용자 계정에 걸쳐 이러한 콘텐츠 아이템을 동기화하는 것은 여전히 기술적인 장애로 결함이 많이 존재한다.

[0007] 기술적 장애 중 일부를 설명하기 위해, 제1 머신(예를 들어, 클라이언트 디바이스 또는 서버)은 사용자가 콘텐츠 관리 시스템에 의해 관리되는 콘텐츠 아이템을 어떻게 수정했는지에 관한 정보를 제공하는 제2 머신으로 통신을 전송할 수 있다. 이러한 통신은 제2 머신에 의해 사용되어 제1 머신 상의 콘텐츠 아이템에 대해 수행된 액션(action)이 제2 머신의 콘텐츠 아이템에 반영되고 제1 머신 상의 콘텐츠 아이템이 제2 머신 상의 콘텐츠 아이템과 실질적으로 동일하게 되도록 제2 머신 상의 콘텐츠 아이템과 동기화될 수 있다.

[0008] 그러나, 여러 개의 통신이 전송될 수 있고, 통신을 전송하는 데 사용된 하나 이상의 네트워크에 의해 사용되는 다양한 네트워크 라우팅 프로토콜, 제1 또는 제2 머신의 기술적 동작, 또는 일부 다른 이유로 인해 이들 통신이 순서에 맞지 않게 수신될 수 있다. 또한, 사용자는 다수의 콘텐츠 아이템에 대해 다수의 수정을 수행하거나, 짧은 시간 내에 이전의 수정을 실행 취소(undo)하거나, 또는 이전에 수정된 콘텐츠 아이템 또는 콘텐츠 아이템 세트에 대한 추가 수정을 신속히 수행할 수 있다. 이것은 이러한 통신이 순서에 맞지 않게 수신되거나, 특정 통신이 오래되었거나, 또는 제2 머신이 최신 상태가 아닌 콘텐츠 아이템에 동작을 수행할 수 있는 가능성을 높인다. 그 결과 많은 동작이 콘텐츠 아이템의 현재 상태와 호환되지 않을 수 있다. 실제로 일부 동작이 다른 동작과 충돌하는지 또는 콘텐츠 아이템의 현재 상태와 충돌하는지 여부를 검출하는 것조차 어려울 수 있다.

[0009] 추가적으로, 동기화 액션에 대한 고유의 대기 시간이 존재한다. 예를 들어, 제1 머신에 취해진 액션은 먼저 제1 머신에 의해 검출되고, 통신이 생성된 다음 네트워크를 통해 전송된다. 통신은 아직 이전의 통신을 처리하고 있을 수 있는 제2 머신에 의해 수신되어, 처리되고, 통신에 상술된 액션이 제2 머신에 취해질 수 있다. 이 예시적인 시나리오에서, 제1 머신, 제2 머신 및 네트워크의 제한된 컴퓨팅 자원(예를 들어, 대역폭, 메모리, 처리 시간, 처리 사이클 등)에 의해 대기 시간이 도입되는 몇 가지 포인트가 있다. 대기 시간이 길어지면 일부 이유로 통신이 콘텐츠 아이템의 현재 상태와 충돌할 가능성이 높아진다. 또한 이러한 충돌된 통신을 처리하고 충돌을 해결하는 것은 처리 시간, 메모리, 에너지 또는 대역폭과 같은 불필요한 컴퓨팅 자원을 소비하여 대기 시간이 더 늘어난다.

[0010] 더 복잡한 문제로서, 콘텐츠 아이템에 액세스하는 제2 머신 및/또는 추가 머신 상의 동일하거나 상이한 사용자가 또한 콘텐츠 아이템에 대한 수정을 수행할 수 있다. 그 결과, 상기 문제가 배가될 수 있으며, 로컬 액션이 원격 액션과 충돌하는지 여부 및/또는 로컬 액션이 최신 콘텐츠 아이템에서 동작하고 있는지 여부에 대한 추가 기술적 문제가 발생한다.

[0011] 개시된 기술은 전술한 기술적 문제뿐만 아니라 다른 기술적 문제에 대한 기술적 해결책을 제공하는 콘텐츠 관리 시스템을 위한 클라이언트 동기화 서비스에 대한 기술의 필요성을 다룬다. 클라이언트 동기화 서비스는 클라이언트 디바이스 상에서 동작하고, 콘텐츠 관리 시스템의 서버 상의 콘텐츠 아이템과 클라이언트 디바이스 상의 대응하는 콘텐츠 아이템 사이의 동기화 불일치를 식별하도록 구성될 수 있다. 각각의 동기화 불일치 시에, 클라이언트 동기화 서비스는 콘텐츠 아이템을 동기화하는데 필요한 동작을 식별하고 이 동작을 개시할 수 있다.

[0012] 클라이언트 동기화 서비스는 서버 상의 콘텐츠 아이템의 상태, 클라이언트 디바이스 상의 콘텐츠 아이템의 상태, 및 트리 데이터 구조 세트("트리")를 이용하여 이들의 동기화 상태를 추적할 수 있다. 일부 실시형태에 따르면, 3개의 트리의 세트가 사용될 수 있다. 3개의 트리는 서버 상태를 나타내는 원격 트리, 클라이언트 디바이스 상의 파일 시스템 상태를 나타내는 로컬 트리, 및 로컬 트리와 원격 트리의 병합 베이스(merge base)를 나타내는 동기 트리를 포함할 수 있다. 병합 베이스는 로컬 트리와 원격 트리의 공통 조상으로 고려되거나 또는 로컬 트리와 원격 트리 사이의 마지막으로 알려진 동기화 상태로 고려될 수 있다. 따라서, 클라이언트 동기화 서비스는 3개의 트리(예를 들어, 원격 트리, 동기 트리 및 로컬 트리)가 모두 동일한 경우 서버 상태와 클라이언트 디바이스 상태가 동기화된 것으로 결정할 수 있다.

[0013] 콘텐츠 아이템의 서버 상태 또는 콘텐츠 아이템의 클라이언트 디바이스 파일 시스템 상태("파일 시스템 상태")에 대한 수정이 검출될 때, 클라이언트 동기화 서비스는 적절한 트리를 업데이트하고 서버 상태와 파일 시스템 상태가 트리의 3원소(triumvirate)에 따라 동기화된다. 트리 중 하나에 대한 업데이트에 따라 서버 상태와 파일 시스템 상태가 동기화되거나 동기화되지 않거나 더 동기화되지 않을 수 있다. 서버 상태와 파일 시스템 상태가 동기화되지 않은 경우, 클라이언트 동기화 서비스는 서버 상태와 파일 시스템 상태를 수렴하고 서버 상태와 파일 시스템 상태를 동기화된 상태에 보다 가까이 가게 하는 데 필요한 적어도 초기 동작 세트를 식별할 수 있다.

[0014] 서버 상태와 파일 시스템 상태를 모니터링하기 위해 트리 데이터 구조 세트에 의존함으로써, 다양한 기술적 문제에 대해 컴퓨팅 기술에 뿌리를 둔 대안 및/또는 해결책을 제공한다. 예를 들어, 클라이언트 동기화 서비스는 파일 상태뿐만 아니라 서버 상태를 추적하고, 두 상태의 병합 베이스의 표현을 저장할 수 있다. 그 결과, 본 기술의 다양한 실시형태는 사용자가 콘텐츠 아이템을 원격으로 수정하는 방법을 지정하고 이러한 수정이 로컬로 구현되는 순서, 수정이 다른 수정과 충돌하는지 또는 오래되었는지 여부, 및 원격 수정이 사용자가 로컬로 수행한 로컬 수정과 충돌하는지 여부를 결정하는 다수의 통신을 수신하는 것과 연관된 기술적 문제를 피한다. 이러한 문제 중 다수는 다른 해결책이 수반된 다양한 행위자(예를 들어, 서버 및 클라이언트 디바이스)의 상태를 추적할 수 없고 상태가 동기화되어 있는지 여부를 신속히 결정할 수 없는 것에서 발생한다. 대신, 이러한 다른 해결책은 서버 상태와 파일 시스템 상태가 동기화되어 있는지 여부에 관계없이 콘텐츠 아이템을 로컬로 수정하는 방법에 대한 지시 사항을 수신하는 것에 의존한다.

[0015] 또한, 서버 상태와 파일 시스템 상태가 지속적으로 모니터링되기 때문에, 이들 상태가 동기화되어 있는지 여부를 결정하는 것은 절차의 복잡성뿐만 아니라 컴퓨팅 시간 및 자원 측면에서 훨씬 더 효율적이다. 아래에 더 상세히 설명되는 바와 같이, 클라이언트 동기화 서비스는 보다 결정론적인 방식으로 서버 상태와 파일 시스템 상태의 증분적이고 체계적인 동기화를 가능하게 한다. 그 결과 콘텐츠 관리 시스템 기능의 확장 및 테스트가 또한 더욱 효율적이 된다.

[0016] 콘텐츠 관리 시스템

[0017] 일부 실시형태에서, 개시된 기술은 무엇보다도 특히 콘텐츠 아이템 동기화 능력 및 협업 특징을 갖는 콘텐츠 관리 시스템의 맥락에서 전개된다. 예시적인 시스템 구성(100)이 클라이언트 디바이스(150)와 상호 작용하는 콘텐츠 관리 시스템(110)을 도시하는 도 1a에 도시되어 있다.

[0018] 계정

[0019] 콘텐츠 관리 시스템(110)은 계정과 연관하여 콘텐츠 아이템을 저장할 수 있을 뿐만 아니라 콘텐츠 아이템(들)을 검색, 수정, 브라우징 및/또는 공유하는 것과 같은 다양한 콘텐츠 아이템 관리 작업을 수행할 수 있다. 또한, 콘텐츠 관리 시스템(110)은 계정이 다수의 클라이언트 디바이스로부터 콘텐츠 아이템(들)에 액세스할 수 있게 한다.

[0020] 콘텐츠 관리 시스템(110)은 복수의 계정을 지원한다. 엔티티(사용자, 사용자 그룹, 팀, 회사 등)는 콘텐츠 관리 시스템으로 계정을 생성할 수 있고 계정의 세부 사항은 계정 데이터베이스(140)에 저장될 수 있다. 계정 데이터베이스(140)는 등록된 엔티티에 대한 프로파일 정보를 저장할 수 있다. 경우에 따라 등록된 엔터티에 대한 프로필 정보는 사용자 이름 및/또는 이메일 주소를 포함한다. 계정 데이터베이스(140)는 계정 유형(예를 들어, 다양한 계층의 무료 또는 유료 계정), 할당된 저장 공간, 사용된 저장 공간, 등록된 콘텐츠 관리 클라이언트 애플리케이션(152)이 상주하는 클라이언트 디바이스(150), 보안 설정, 개인 구성 설정 등과 같은 계정 관리 정보를 포함할 수 있다.

[0021] 계정 데이터베이스(140)는 엔티티와 연관된 계정 그룹을 저장할 수 있다. 그룹은 그룹 정책 및/또는 액세스 제어 리스트에 기초하여 권한(permission)을 가질 수 있으며, 그룹의 구성원은 권한을 상속할 수 있다. 예를 들어, 마케팅 그룹은 하나의 세트의 콘텐츠 아이템에 액세스할 수 있는 반면, 엔지니어링 그룹은 다른 세트의 콘텐츠 아이템에 액세스할 수 있다. 관리자 그룹은 그룹을 수정할 수 있고, 사용자 계정을 수정할 수 있고, 다른 것을 수정할 수 있다.

[0022] 콘텐츠 아이템 저장

[0023] 콘텐츠 관리 시스템(110)의 특징은 콘텐츠 저장소(142)에 저장될 수 있는 콘텐츠 아이템을 저장하는 것이다. 콘텐츠 아이템은 문서, 협업 콘텐츠 아이템, 텍스트 파일, 오디오 파일, 이미지 파일, 비디오 파일, 웹 페이지, 실행 파일, 이진 파일 등과 같은 임의의 디지털 데이터일 수 있다. 콘텐츠 아이템에는 폴더, 집(zip) 파일, 재생 리스트, 앨범 등과 같은 다른 거동과 함께 콘텐츠 아이템을 그룹화하기 위한 콜렉션(collection) 또는 다른 메커니즘이 포함될 수 있다. 콜렉션은 폴더, 또는 공통 속성에 의해 그룹화되거나 연관된 복수의 콘텐츠 아이템을 지칭할 수 있다. 일부 실시형태에서, 콘텐츠 저장소(142)는 특정 기능을 처리하기 위해 다른 유형의 저장소 또는 데이터베이스와 결합된다. 콘텐츠 저장소(142)는 콘텐츠 아이템을 저장할 수 있는 반면, 콘텐츠 아이템에 관한 메타데이터는 메타데이터 데이터베이스(146)에 저장될 수 있다. 마찬가지로, 콘텐츠 아이템이 콘텐츠 저장소(142)에 저장된 위치에 관한 데이터는 콘텐츠 디렉토리(144)에 저장될 수 있다. 추가적으로 변경, 액세스 등에 관한 데이터는 서버 파일 저널(148)에 저장될 수 있다. 콘텐츠 저장소(142), 콘텐츠 디렉토리(144), 서버 파일 저널(148) 및 메타데이터 데이터베이스(146)와 같은 다양한 저장소/데이터베이스 각각은 하나를 초과하는 이러한 저장소 또는 데이터베이스로 구성될 수 있고, 많은 디바이스 및 위치에 걸쳐 분산될 수 있다. 다른 구성도 가능하다. 예를 들어, 콘텐츠 저장소(142), 콘텐츠 디렉토리(144), 서버 파일 저널(148) 및/또는 메타데이터 데이터베이스(146)로부터의 데이터는 하나 이상의 콘텐츠 저장소 또는 데이터베이스로 결합되거나, 추가적인 콘텐츠 저장소 또는 데이터베이스로 더 분할될 수 있다. 따라서, 콘텐츠 관리 시스템(110)은 도 1에 도시된 것보다 더 많거나 더 적은 저장소 및/또는 데이터베이스를 포함할 수 있다.

[0024] 일부 실시형태에서, 콘텐츠 저장소(142)는 저장을 위한 콘텐츠 아이템의 수신, 저장을 위한 콘텐츠 아이템의 준비, 콘텐츠 아이템을 위한 저장 위치의 선택, 저장으로부터 콘텐츠 아이템의 검색 등을 포함하지만 이로 제한되지 않는 콘텐츠 아이템의 저장을 관리하기 위한 소프트웨어 또는 다른 프로세서 실행 가능 명령을 포함하는 적어도 하나의 콘텐츠 저장 서비스(116)와 연관된다. 일부 실시형태에서, 콘텐츠 저장 서비스(116)는 콘텐츠 아이템을 콘텐츠 저장소(142)에 저장하기 위해 더 작은 청크(chunk)로 분할할 수 있다. 콘텐츠 아이템을 구성하는 각 청크의 위치는 콘텐츠 디렉토리(144)에 기록될 수 있다. 콘텐츠 디렉토리(144)는 콘텐츠 저장소(142)에 저장된 각 콘텐츠 아이템에 대한 콘텐츠 엔트리를 포함할 수 있다. 콘텐츠 엔트리는 콘텐츠 아이템을 식별하는 고유 ID와 연관될 수 있다.

[0025] 일부 실시형태에서, 콘텐츠 디렉토리(144)에서 콘텐츠 아이템을 식별하는 고유 ID는 결정론적 해시 함수(hash function)로부터 도출될 수 있다. 콘텐츠 아이템에 대한 고유 ID를 도출하는 이 방법은 결정론적 해시 함수가 동일한 콘텐츠 아이템의 모든 사본에 대해 동일한 식별자를 출력하지만, 다른 콘텐츠 아이템에 대해 다른 식별자를 출력하기 때문에 콘텐츠 아이템 복제본이 인식되는 것을 보장할 수 있다. 이 방법을 사용하여, 콘텐츠 저장 서비스(116)는 각 콘텐츠 아이템에 대한 고유 ID를 출력할 수 있다.

[0026] 콘텐츠 저장 서비스(116)는 또한 메타데이터 데이터베이스(146)에서 콘텐츠 아이템에 대한 콘텐츠 경로를 지정하거나 기록할 수 있다. 콘텐츠 경로는 콘텐츠 아이템의 이름 및/또는 콘텐츠 아이템과 연관된 폴더 계층을 포함할 수 있다. 예를 들어, 콘텐츠 경로는 콘텐츠 아이템이 클라이언트 디바이스 상의 로컬 파일 시스템에 저장되는 폴더를 포함하거나 또는 폴더들의 경로를 포함할 수 있다. 콘텐츠 아이템은 콘텐츠 저장소(142)에 블록으로 저장되고 트리와 같은 디렉토리 구조 아래에 저장되지 않을 수 있지만, 이러한 디렉토리 구조는 사용자에게 편안한 내비게이션 구조이다. 콘텐츠 저장 서비스(116)는 콘텐츠 아이템에 대한 콘텐츠 경로를 규정하거나 기록

할 수 있으며, 여기서 디렉토리 구조의 "루트(root)" 노드는 각 계정의 네임스페이스일 수 있다. 네임스페이스 내에는 계정 및/또는 콘텐츠 저장 서비스(116)의 사용자에 의해 규정된 디렉토리 구조일 수 있다. 메타데이터 데이터베이스(146)는 각 콘텐츠 아이템에 대한 콘텐츠 경로를 콘텐츠 엔트리의 일부로서 저장할 수 있다.

[0027] 일부 실시형태에서, 네임스페이스는 마치 루트 노드 내에 저장된 것처럼 디렉토리 구조에 내포된 추가 네임스페이스를 포함할 수 있다. 이것은 계정이 공유 콜렉션에 액세스할 수 있는 경우에 발생할 수 있다. 공유 콜렉션은 콘텐츠 관리 시스템(110) 내에서 자신의 네임스페이스를 할당받을 수 있다. 일부 공유 콜렉션은 실제로는 공유 콜렉션의 루트 노드이지만 이 공유 콜렉션은 디렉토리 구조에서 계정 네임스페이스의 하위에 위치되어 계정을 위한 폴더 내에 폴더로 나타날 수 있다. 위에서 언급된 바와 같이, 디렉토리 구조는 단지 사용자에 편안한 내비게이션 구조이지만, 콘텐츠 저장소(142)에서 콘텐츠 아이템의 저장 위치와 상관된 것은 아니다.

[0028] 계정이 콘텐츠 아이템을 보는 디렉토리 구조가 콘텐츠 관리 시스템(110)에서의 저장 위치와 상관되지 않지만, 디렉토리 구조는 클라이언트 디바이스(150)에 의해 사용되는 파일 시스템에 따라 클라이언트 디바이스(150) 상의 저장 위치와 상관될 수 있다.

[0029] 전술한 바와 같이, 콘텐츠 디렉토리(144)의 콘텐츠 엔트리는 또한 콘텐츠 아이템을 구성하는 각 청크의 위치를 포함할 수 있다. 보다 구체적으로, 콘텐츠 엔트리는 콘텐츠 아이템을 구성하는 청크의 콘텐츠 저장소(142)에서의 위치를 식별하는 콘텐츠 포인터를 포함할 수 있다.

[0030] 콘텐츠 경로 및 콘텐츠 포인터에 더하여, 콘텐츠 디렉토리(144)의 콘텐츠 엔트리는 또한 콘텐츠 아이템에 액세스를 하는 사용자 계정을 식별하는 사용자 계정 식별자, 및/또는 콘텐츠 엔트리가 속하는 네임스페이스 및/또는 콘텐츠 아이템에 액세스를 하는 그룹을 식별하는 그룹 식별자를 포함할 수 있다.

[0031] 콘텐츠 저장 서비스(116)는 콘텐츠 아이템 또는 콘텐츠 아이템의 버전(version)을 구성하는 중복 콘텐츠 아이템 또는 중복 블록을 식별함으로써 필요한 저장 공간의 양을 감소시킬 수 있다. 다수의 사본을 저장하는 대신에, 콘텐츠 저장소(142)는 콘텐츠 아이템 또는 콘텐츠 아이템의 블록의 단일 사본을 저장할 수 있고, 콘텐츠 디렉토리(144)는 복제물을 단일 사본에 링크하기 위한 포인터 또는 다른 메커니즘을 포함할 수 있다.

[0032] 콘텐츠 저장 서비스(116)는 또한 콘텐츠 아이템의 고유 ID와 연관하여, 콘텐츠 아이템, 콘텐츠 아이템 유형, 폴더, 파일 경로를 기술하는 메타데이터, 및/또는 메타데이터 데이터베이스(146)의 다양한 계정, 콜렉션 또는 그룹과 콘텐츠 아이템의 관계를 저장할 수 있다.

[0033] 콘텐츠 저장 서비스(116)는 또한 서버 파일 저널(148)에 변경, 액세스 등에 관한 데이터의 로그(log)를 저장할 수 있다. 서버 파일 저널(148)은 콘텐츠 아이템의 고유 ID, 및 타임 스탬프 또는 버전 번호와 함께 변경 사항 또는 액세스 액션에 대한 설명, 및 임의의 다른 관련 데이터를 포함할 수 있다. 서버 파일 저널(148)은 또한 변경 또는 콘텐츠 아이템 액세스에 의해 영향을 받는 블록에 대한 포인터를 포함할 수 있다. 콘텐츠 저장 서비스는 콘텐츠 아이템의 변경 사항, 콘텐츠 아이템의 다른 버전(분기하는 버전 트리를 포함), 및 서버 파일 저널(148)로부터 얻을 수 있는 변경 이력을 추적하는 콘텐츠 아이템 버전 제어를 사용하여 동작을 실행 취소할 수 있는 기능을 제공할 수 있다.

[0034] 콘텐츠 아이템 동기화

[0035] 콘텐츠 관리 시스템(110)의 다른 특징은 콘텐츠 아이템을 적어도 하나의 클라이언트 디바이스(150)와 동기화하는 것이다. 클라이언트 디바이스(들)는 상이한 형태를 취할 수 있고 상이한 능력을 가질 수 있다. 예를 들어, 클라이언트 디바이스(150_1)는 상주하는 다수의 애플리케이션에 의해 액세스 가능한 로컬 파일 시스템을 갖는 컴퓨팅 디바이스이다. 클라이언트 디바이스(150_2)는 콘텐츠 아이템이 특정 애플리케이션에 의해서만 액세스 가능하거나 또는 특정 애플리케이션에 의해 주어진 권한에 의해서만 액세스 가능하고, 콘텐츠 아이템은 전형적으로 애플리케이션 특정 공간 또는 클라우드에 저장되는 컴퓨팅 디바이스이다. 클라이언트 디바이스(150_3)는 웹 브라우저를 통해 콘텐츠 관리 시스템(110)에 액세스하고 웹 인터페이스를 통해 콘텐츠 아이템에 액세스하는 임의의 클라이언트 디바이스이다. 예시적인 클라이언트 디바이스(150_1, 150_2 및 150_3)는 랩탑, 모바일 디바이스 또는 웹 브라우저와 같은 폼 팩터로 도시되어 있지만, 이러한 설명은 이러한 예시적인 폼 팩터의 디바이스로 제한되지 않는 것으로 이해된다. 예를 들어, 클라이언트(150_2)와 같은 모바일 디바이스는 상주하는 다수의 애플리케이션에 의해 액세스 가능한 로컬 파일 시스템을 가질 수 있고, 또는 클라이언트(150_2)는 웹 브라우저를 통해 콘텐츠 관리 시스템(110)에 액세스할 수 있다. 따라서 클라이언트(150)의 능력을 고려할 때 폼 팩터는 본 발명을 제한하

는 것으로 고려되어서는 안 된다. 클라이언트 디바이스(150)와 관련하여 본 명세서에 설명된 하나 이상의 기능은 디바이스의 특정 능력에 따라 모든 클라이언트 디바이스에서 이용 가능하거나 가능하지 않을 수 있고, 파일 액세스 모델은 이러한 능력 중 하나이다.

[0036] 많은 실시형태에서, 클라이언트 디바이스는 콘텐츠 관리 시스템(110)의 계정과 연관되지만, 일부 실시형태에서 클라이언트 디바이스는 공유 링크를 사용하여 콘텐츠에 액세스할 수 있고 계정을 요구하지 않는다.

[0037] 위에서 언급한 바와 같이, 일부 클라이언트 디바이스는 웹 브라우저를 사용하여 콘텐츠 관리 시스템(110)에 액세스할 수 있다. 그러나, 클라이언트 디바이스는 또한 클라이언트 디바이스(150)에 저장되고 클라이언트 디바이스에서 실행되는 클라이언트 애플리케이션(152)을 사용하여 콘텐츠 관리 시스템(110)에 액세스할 수 있다. 클라이언트 애플리케이션(152)은 클라이언트 동기화 서비스(156)를 포함할 수 있다.

[0038] 클라이언트 동기화 서비스(156)는 서버 동기화 서비스(112)와 통신하며 클라이언트 디바이스(150)와 콘텐츠 관리 시스템(110) 사이의 콘텐츠 아이템에 대한 변경을 동기화할 수 있다.

[0039] 클라이언트 디바이스(150)는 클라이언트 동기화 서비스(156)를 통해 콘텐츠 관리 시스템(110)과 콘텐츠를 동기화할 수 있다. 동기화는 플랫폼에 구애받지 않을 수 있다. 즉, 콘텐츠는 다양한 유형, 능력, 동작 시스템 등의 다수의 클라이언트 디바이스에 걸쳐 동기화될 수 있다. 클라이언트 동기화 서비스(156)는 임의의 변경(신규, 삭제, 수정, 복사 또는 이동된 콘텐츠 아이템)을 클라이언트 디바이스(150)의 파일 시스템의 지정된 위치의 콘텐츠 아이템과 동기화할 수 있다.

[0040] 콘텐츠 아이템은 클라이언트 디바이스(150)로부터 콘텐츠 관리 시스템(110)과 동기화될 수 있으며, 그 역도 마찬가지이다. 클라이언트 디바이스(150)로부터 콘텐츠 관리 시스템(110)과 동기화되는 실시형태에서, 사용자는 클라이언트 디바이스(150)의 파일 시스템으로부터 직접 콘텐츠 아이템을 조작할 수 있는 반면, 클라이언트 동기화 서비스(156)는 클라이언트 디바이스(150) 상의 디렉토리에서 모니터링되는 폴더 내 파일에 대한 변경 사항을 모니터링할 수 있다.

[0041] 클라이언트 동기화 서비스(156)는 모니터링하는 디렉토리에서 콘텐츠의 기입, 이동, 복사 또는 삭제를 검출할 때, 클라이언트 동기화 서비스(156)는 변경 사항을 콘텐츠 관리 시스템 서비스(116)와 동기화할 수 있다. 일부 실시형태에서 콘텐츠 동기화 시스템(156)은 콘텐츠 아이템을 블록으로 분할하고, 콘텐츠 아이템을 해싱하여 고유 식별자를 생성하는 등을 포함하여, 위에서 언급된 기능을 포함하여 콘텐츠 관리 시스템 서비스(116)의 일부 기능을 수행할 수 있다. 클라이언트 동기화 서비스(156)는 클라이언트 저장 인덱스(164)에서 콘텐츠를 인덱싱하고 결과를 저장 인덱스(164)에 저장할 수 있다. 인덱싱은 저장 경로에 더하여 고유 서버 식별자, 및 각 콘텐츠 아이템에 대한 고유 클라이언트 식별자를 포함할 수 있다. 일부 실시형태에서, 클라이언트 동기화 서비스(156)는 서버 동기화 서비스(112)로부터 고유 서버 식별자를 학습하고, 클라이언트 디바이스(150)의 동작 시스템으로부터 고유 클라이언트 식별자를 학습한다.

[0042] 클라이언트 동기화 서비스(156)는 저장 인덱스(164)를 사용하여 클라이언트 저장소 내의 콘텐츠의 적어도 일부와 콘텐츠 관리 시스템(110) 상의 사용자 계정과 연관된 콘텐츠와 동기화를 용이하게 할 수 있다. 예를 들어, 클라이언트 동기화 서비스(156)는 저장 인덱스(164)를 콘텐츠 관리 시스템(110)과 비교하고, 콘텐츠 관리 시스템(110) 상의 사용자 계정과 연관된 콘텐츠와 클라이언트 저장소 상의 콘텐츠 사이의 차이를 검출할 수 있다. 클라이언트 동기화 서비스(156)는 적절한 경우 클라이언트 저장소 상의 콘텐츠를 업로드, 다운로드, 수정 및 삭제함으로써 차이를 조정하려고 시도할 수 있다. 콘텐츠 저장 서비스(116)는 콘텐츠 아이템에 대한 변경된 또는 새로운 블록을 저장하고, 적절한 경우 서버 파일 저널(148), 메타데이터 데이터베이스(146), 콘텐츠 디렉토리(144), 콘텐츠 저장소(142), 계정 데이터베이스(140) 등을 업데이트할 수 있다.

[0043] 콘텐츠 관리 시스템(110)으로부터 클라이언트 디바이스(150)로 동기화할 때, 서버 파일 저널(148)에 기록된 콘텐츠 아이템의 마운트, 수정, 추가, 삭제, 이동은 통지 서비스(117)를 사용하여 클라이언트 디바이스(150)로 통지가 전송되도록 트리거할 수 있다. 클라이언트 디바이스(150)가 변경 사항을 통지받으면, 클라이언트 디바이스에 알려진 마지막 동기화 지점 이후 요청 변경 사항이 서버 파일 저널(148)에 나열된다. 클라이언트 디바이스(150)가 콘텐츠 관리 시스템(110)과 동기화되지 않은 것으로 결정하면, 클라이언트 동기화 서비스(156)는 변경 사항을 포함하는 콘텐츠 아이템 블록을 요청하고, 변경된 콘텐츠 아이템의 로컬 사본을 업데이트한다.

[0044] 일부 실시형태에서, 저장 인덱스(164)는 트리 데이터 구조를 저장하고, 하나의 트리는 서버 동기화 서비스(112)에 따라 디렉토리의 최신 표현을 반영하는 반면, 다른 트리는 클라이언트 동기화 서비스(156)에 따른 디렉토리의 최신 표현을 반영한다. 클라이언트 동기화 서비스는 서버 동기화 서비스(112)로부터 데이터를 요청하거나

클라이언트 디바이스(150) 상의 변경 사항을 콘텐츠 관리 시스템(110)에 커밋(commit)함으로써 트리 구조가 일치하는 것을 보장하도록 작동할 수 있다.

[0045] 때때로 클라이언트 디바이스(150)는 이용 가능한 네트워크 연결을 갖지 않을 수 있다. 이 시나리오에서, 클라이언트 동기화 서비스(156)는 콘텐츠 아이템을 변경하기 대한 링크된 콜렉션을 모니터링하고, 네트워크 연결이 이용 가능할 때 나중에 콘텐츠 관리 시스템(110)과 동기화하기 위해 이러한 변경을 대기시킬 수 있다. 유사하게, 사용자는 콘텐츠 관리 시스템(110)과의 동기화를 수동으로 시작, 중지, 일시 정지 또는 재개할 수 있다.

[0046] 클라이언트 동기화 서비스(156)는 콘텐츠 관리 시스템(110) 상의 특정 사용자 계정과 연관된 모든 콘텐츠를 동기화할 수 있다. 대안적으로, 클라이언트 동기화 서비스(156)는 콘텐츠 관리 시스템(110) 상의 특정 사용자 계정과 연관된 전체 콘텐츠의 콘텐츠의 일부를 선택적으로 동기화할 수 있다. 콘텐츠의 일부만을 선택적으로 동기화하는 것은 클라이언트 디바이스(150) 상의 공간을 보존하고 대역폭을 절약할 수 있다.

[0047] 일부 실시형태에서, 클라이언트 동기화 서비스(156)는 특정 사용자 계정과 연관된 콘텐츠의 일부를 선택적으로 저장하고, 콘텐츠의 나머지 부분을 위한 클라이언트 저장소에 플레이스홀더 콘텐츠 아이템을 저장한다. 예를 들어, 클라이언트 동기화 서비스(156)는 콘텐츠 관리 시스템(110) 상에 각각의 완전한 콘텐츠 아이템의, 동일한 파일명, 경로, 확장자, 메타데이터를 갖지만 완전한 콘텐츠 아이템의 데이터는 없는 플레이스홀더 콘텐츠 아이템을 저장할 수 있다. 플레이스홀더 콘텐츠 아이템은 수 바이트 이하 크기일 수 있지만 각각의 완전한 콘텐츠 아이템은 상당히 클 수 있다. 클라이언트 디바이스(150)가 콘텐츠 아이템에 액세스를 시도한 후에, 클라이언트 동기화 서비스(156)는 콘텐츠 관리 시스템(110)으로부터 콘텐츠 아이템의 데이터를 검색하고, 액세스하는 클라이언트 디바이스(150)에 완전한 콘텐츠 아이템을 제공할 수 있다. 이 접근법은 콘텐츠 관리 시스템(110) 상의 사용자의 콘텐츠에 여전히 완전한 액세스를 제공하면서 상당한 공간 및 대역폭 절약을 제공할 수 있다.

[0048] 협업 특징

[0049] 콘텐츠 관리 시스템(110)의 다른 특징은 사용자들 간의 협업을 수행하는 것이다. 협업 특징은 콘텐츠 아이템의 공유, 콘텐츠 아이템에 코멘트 달기(commenting), 콘텐츠 아이템에 공동 작업, 인스턴트 메시징, 콘텐츠 아이템에 관한 존재 및 보여지는 상태 정보의 제공 등을 포함한다.

[0050] 공유

[0051] 콘텐츠 관리 시스템(110)은 공유 서비스(128)를 통해 공유 콘텐츠를 관리할 수 있다. 콘텐츠에 대한 링크를 제공함으로써 콘텐츠를 공유하는 것은 콘텐츠 관리 시스템(110)과 네트워크 통신하는 임의의 컴퓨팅 디바이스로부터 콘텐츠 아이템에 액세스 가능하게 하는 것을 포함할 수 있다. 그러나, 일부 실시형태에서 링크는 콘텐츠 관리 시스템(110) 및 액세스 제어 리스트(145)에 의해 시행되는 액세스 제한과 연관될 수 있다. 콘텐츠를 공유하는 것은 또한 공유 서비스(128)를 사용하여 콘텐츠를 링크하여 (콘텐츠 아이템과 연관된 원래 사용자 계정에 더하여) 적어도 하나의 추가 사용자 계정과 콘텐츠 관리 시스템(110) 내 콘텐츠를 공유하여 각 사용자 계정이 콘텐츠 아이템에 액세스할 수 있도록 하는 것을 포함할 수 있다. 추가 사용자 계정은 콘텐츠를 수락함으로써 콘텐츠에 대한 액세스를 얻을 수 있고, 이 콘텐츠는 웹 인터페이스 서비스(124)를 통해 액세스되거나 또는 클라이언트 디바이스(150) 상의 이들의 계정과 연관된 디렉토리 구조 내로부터 직접 액세스될 수 있다. 공유는 플랫폼에 구애됨이 없이 수행될 수 있다. 즉, 콘텐츠는 다양한 유형, 능력, 동작 시스템 등의 다수의 클라이언트 디바이스(150)에 걸쳐 공유될 수 있다. 콘텐츠는 또한 다양한 유형의 사용자 계정에 걸쳐 공유될 수 있다.

[0052] 콘텐츠 관리 시스템(110) 내 콘텐츠 아이템을 공유하기 위해 공유 서비스(128)는 콘텐츠 아이템과 연관된 액세스 제어 리스트 데이터베이스(145)의 콘텐츠 엔트리에 사용자 계정 식별자 또는 다수의 사용자 계정 식별자를 추가하여 콘텐츠 아이템에 대해 추가된 사용자 계정의 액세스를 부여할 수 있다. 공유 서비스(128)는 또한 콘텐츠 아이템에 대한 사용자 계정의 액세스를 제한하기 위해 콘텐츠 엔트리로부터 사용자 계정 식별자를 제거할 수 있다. 공유 서비스(128)는 콘텐츠 아이템 식별자, 콘텐츠 아이템에 대한 액세스가 주어진 사용자 계정 식별자, 및 액세스 레벨을 액세스 제어 리스트 데이터베이스(145)에 기록할 수 있다. 예를 들어, 일부 실시형태에서, 단일 콘텐츠 엔트리와 연관된 사용자 계정 식별자는 연관된 콘텐츠 아이템에 대한 각각의 사용자 계정 식별자의 상이한 권한을 지정할 수 있다.

[0053] 콘텐츠 관리 시스템(110) 외부에서 콘텐츠 아이템을 공유하기 위해, 공유 서비스(128)는 어떤 인증도 없이 어떤 웹 브라우저라도 콘텐츠 관리 시스템(110) 내 콘텐츠 아이템 또는 콜렉션에 액세스할 수 있게 하는 범용 자원 할당자(Uniform Resource Locator: URL)와 같은 커스텀 네트워크 주소를 생성할 수 있다. 이를 달성하기 위해, 공유 서비스(128)는 생성된 URL에 콘텐츠 식별 데이터를 포함할 수 있으며, 이는 요청된 콘텐츠 아이템을 적절

히 식별하고 리턴하기 위해 나중에 사용될 수 있다. 예를 들어, 공유 서비스(128)는 생성된 URL 내에 계정 식별자 및 콘텐츠 경로 또는 콘텐츠 아이템 식별 코드를 포함할 수 있다. URL을 선택하면, URL에 포함된 콘텐츠 식별 데이터는 콘텐츠 관리 시스템(110)으로 전송될 수 있으며, 콘텐츠 관리 시스템은 수신된 콘텐츠 식별 데이터를 사용하여 적절한 콘텐츠 아이템을 식별하고 콘텐츠 아이템을 반환할 수 있다.

[0054] URL을 생성하는 것에 더하여, 공유 서비스(128)는 또한 콘텐츠 아이템에 대한 URL이 생성되었음을 액세스 제어 리스트 데이터베이스(145)에 기록하도록 구성될 수 있다. 일부 실시형태에서, 콘텐츠 아이템과 연관된 콘텐츠 엔트리는 콘텐츠 아이템에 대한 URL이 생성되었는지 여부를 나타내는 URL 플래그를 포함할 수 있다. 예를 들어, URL 플래그는 콘텐츠 아이템에 대한 URL이 생성되지 않았다는 것을 나타내기 위해 초기에 0 또는 거짓으로 설정된 부울리안 값일 수 있다. 공유 서비스(128)는 콘텐츠 아이템에 대한 URL을 생성한 후 플래그의 값을 1 또는 참으로 변경할 수 있다.

[0055] 일부 실시형태에서, 공유 서비스(128)는 권한 세트를 콘텐츠 아이템에 대한 URL에 연관시킬 수 있다. 예를 들어, 사용자가 URL을 통해 콘텐츠 아이템에 액세스하려고 하면, 공유 서비스(128)는 콘텐츠 아이템에 대해 제한된 권한 세트를 제공할 수 있다. 제한된 권한의 예로는 사용자가 콘텐츠 아이템을 다운로드할 수 없다는 것, 콘텐츠 아이템을 저장할 수 없다는 것, 콘텐츠 아이템을 복사할 수 없다는 것, 콘텐츠 아이템을 수정할 수 없다는 것 등의 제한을 포함한다. 일부 실시형태에서, 제한된 권한은 콘텐츠 아이템이 특정 도메인과만, 즉 회사 네트워크 도메인으로부터만 액세스되거나 또는 특정 도메인과 연관된 계정에 의해서만, 예를 들어, 회사 계정과 연관된 계정(예를 들어, @acme.com)에 의해서만 액세스될 수 있게 하는 제한을 포함한다.

[0056] 일부 실시형태에서, 공유 서비스(128)는 생성된 URL을 비활성화하도록 더 구성될 수 있다. 예를 들어, 각 콘텐츠 엔트리는 또한 생성된 URL로부터의 요청에 응답하여 콘텐츠가 반환되어야 하는지 여부를 나타내는 URL 활성 플래그를 포함할 수 있다. 예를 들어, 공유 서비스(128)는 URL 활성 플래그가 1 또는 참으로 설정된 경우에만 생성된 링크에 의해 요청된 콘텐츠 아이템을 반환할 수 있다. 따라서, URL 활성 플래그의 값을 변경함으로써 URL이 생성된 콘텐츠 아이템에 대한 액세스를 용이하게 제한할 수 있다. 이를 통해 사용자는 콘텐츠 아이템을 이동시키거나 생성된 URL을 삭제하지 않고도 공유 콘텐츠 아이템에 대한 액세스를 제한할 수 있다. 마찬가지로, 공유 서비스(128)는 URL 활성 플래그의 값을 1 또는 참으로 다시 변경함으로써 URL을 재활성화할 수 있다. 따라서 사용자는 새로운 URL을 생성하지 않고도 콘텐츠 아이템에 대한 액세스를 쉽게 복원할 수 있다.

[0057] 일부 실시형태에서, 콘텐츠 관리 시스템(110)은 콘텐츠 아이템을 업로드하기 위한 URL을 지정할 수 있다. 예를 들어, 사용자 계정이 있는 제1 사용자는 이 URL을 요청하고, 기여하는 사용자에게 URL을 제공할 수 있고, 기여하는 사용자는 이 URL을 사용하여 제1 사용자의 사용자 계정에 콘텐츠 아이템을 업로드할 수 있다.

[0058] 팀 서비스

[0059] 일부 실시형태에서, 콘텐츠 관리 시스템(110)은 팀 서비스(130)를 포함한다. 팀 서비스(130)는 사용자 계정의 정해진 팀을 생성 및 관리하기 위한 기능을 제공할 수 있다. 팀은 회사에 대해 서브 팀(예를 들어, 사업부 또는 프로젝트 팀 등)이 있는 것으로 생성될 수 있고, 팀 및 서브 팀에 할당된 사용자 계정은 임의의 정해진 사용자 계정 그룹에 대해 생성될 수 있다. 팀 서비스(130)는 팀을 위한 공통 공유 공간, 개인 사용자 계정 폴더, 및 액세스 제한된 공유 폴더를 제공할 수 있다. 또한 팀 서비스는 관리자가 팀 내의 콜렉션과 콘텐츠 아이템을 관리하기 위한 관리 인터페이스를 제공할 수 있고, 팀과 연관된 사용자 계정을 관리할 수 있다.

[0060] 권한 부여 서비스(authorization service)

[0061] 일부 실시형태에서, 콘텐츠 관리 시스템(110)은 권한 부여 서비스(132)를 포함한다. 권한 부여 서비스(132)는 네임스페이스에 액세스를 시도하는 사용자 계정이 네임스페이스에 액세스하기 위한 적절한 권리를 갖는 것을 보장한다. 권한 부여 서비스(132)는 네임스페이스에 액세스하는 요청을 따르는 토큰을 클라이언트 애플리케이션(152)으로부터 수신할 수 있고 사용자 계정에 허용된 능력을 리턴할 수 있다. 사용자 계정이 다수의 액세스 레벨을 갖는 경우(예를 들어, 사용자 계정이 사용자 권리 및 관리자 권리를 갖는 경우) 권한 부여 서비스(132)는 또한 관리자에 의한 의도치 않은 액션을 피하기 위해 명시적인 특권 상승(privilege escalation)을 요구할 수 있다.

[0062] 존재 및 보여지는 상태

[0063] 일부 실시형태에서, 콘텐츠 관리 시스템은 콘텐츠 아이템이 공유되는 사용자들이 콘텐츠 아이템과 어떻게 상호작용하는지 또는 상호 작용했는지에 관한 정보를 제공할 수 있다. 일부 실시형태에서, 콘텐츠 관리 시스템(110)은 콘텐츠 아이템을 공유하는 사용자가 콘텐츠 아이템을 현재 보고 있다고 보고할 수 있다. 예를 들어, 클라

이언트 협업 서비스(160)는 클라이언트 디바이스(150)가 콘텐츠 아이템에 액세스하고 있을 때 통지 서비스(117)에 통지할 수 있다. 통지 서비스(117)는 다른 사용자가 콘텐츠 아이템에 대하여 클라이언트 디바이스(150)의 사용자의 존재의 동일한 콘텐츠 아이템에 액세스하는 것을 모든 클라이언트 디바이스에 통지할 수 있다.

[0064] 일부 실시형태에서, 콘텐츠 관리 시스템(110)은 공유 콘텐츠 아이템과 사용자의 상호 작용의 이력을 보고할 수 있다. 협업 서비스(126)는 메타데이터 데이터베이스(146) 및 서버 파일 저널(148)과 같은 데이터 소스(data source)를 조회하여, 사용자가 콘텐츠 아이템을 저장했는지, 사용자가 아직 콘텐츠 아이템을 보지 않았는지 등을 결정하고, 통지 서비스(117)를 사용하여 이 상태 정보를 다른 사용자에 분배하여, 다른 사용자들이 콘텐츠 아이템을 현재 보고 있거나 보았거나 또는 수정하고 있거나 수정한 사람을 알 수 있도록 할 수 있다.

[0065] 협업 서비스(126)는 콘텐츠 아이템이 기본적으로 코멘트 달기 기능을 지원하지 않는다 하더라도 콘텐츠와 연관된 코멘트를 수행할 수 있다. 이러한 코멘트는 메타데이터 데이터베이스(146)에 저장될 수 있다.

[0066] 협업 서비스(126)는 사용자에 대한 통지를 발신하고 전송할 수 있다. 예를 들어, 사용자는 코멘트에서 다른 사용자를 언급할 수 있고, 협업 서비스(126)는 이 사용자가 코멘트에서 언급되었다는 통지를 이 사용자에 전송할 수 있다. 다양한 다른 콘텐츠 아이템 이벤트는 콘텐츠 아이템 삭제, 콘텐츠 아이템 공유 등을 포함하여 통지를 트리거할 수 있다.

[0067] 협업 서비스(126)는 사용자가 인스턴트 메시지, 음성 통화, 이메일 등을 송수신할 수 있는 메시징 플랫폼을 제공할 수 있다.

[0068] 협업 콘텐츠 아이템

[0069] 일부 실시형태에서, 콘텐츠 관리 서비스는, 사용자들이 협업 콘텐츠 아이템을 동시에 생성하고, 협업 콘텐츠 아이템에 코멘트를 달고, 협업 콘텐츠 아이템 내의 작업을 관리할 수 있는 대화식 콘텐츠 아이템 협업 플랫폼을 제공할 수 있는 협업 문서 서비스(134)를 더 포함할 수 있다. 협업 콘텐츠 아이템은 사용자가 협업 콘텐츠 아이템 편집기를 사용하여 생성 및 편집할 수 있는 파일일 수 있으며, 협업 콘텐츠 아이템 요소를 포함할 수 있다. 협업 콘텐츠 아이템 요소는 협업 콘텐츠 아이템 식별자, 하나 이상의 저자 식별자, 협업 콘텐츠 아이템 텍스트, 협업 콘텐츠 아이템 속성, 상호 작용 정보, 코멘트, 공유 사용자 등을 포함할 수 있다. 협업 콘텐츠 아이템 요소는 협업 콘텐츠 아이템을 탐색하고 검색할 수 있게 하는 데이터베이스 엔티티로서 저장될 수 있다. 여러 사용자가 동시에 또는 다른 시간에 협업 콘텐츠 아이템에 액세스하고 보고 편집하고 협업할 수 있다. 일부 실시형태에서, 이것은 2명의 사용자가 웹 인터페이스를 통해 콘텐츠 아이템에 액세스할 것을 요구함으로써 관리될 수 있고, 거기서 이들 사용자는 콘텐츠 아이템의 동일한 사본에서 동시에 작업할 수 있다.

[0070] 협업 컴패니언(companion) 인터페이스

[0071] 일부 실시형태에서, 클라이언트 협업 서비스(160)는 클라이언트 디바이스(150) 상에 제시되는 콘텐츠 아이템과 관련된 정보를 디스플레이하기 위한 기본 애플리케이션 컴패니언 인터페이스를 제공할 수 있다. 콘텐츠 아이템이 클라이언트 디바이스(150)에 저장되고 클라이언트 디바이스에서 실행되는 기본 애플리케이션에 의해 액세스되고, 콘텐츠 아이템이 콘텐츠 애플리케이션(152)에 의해 관리되도록 콘텐츠 아이템이 클라이언트 디바이스(150)의 파일 시스템의 지정된 위치에 있는 실시형태에서, 기본 애플리케이션은 상기 언급된 협업 데이터를 디스플레이하는 어떤 기본 방식도 제공하지 않을 수 있다. 이러한 실시형태에서, 클라이언트 협업 서비스(160)는 사용자가 콘텐츠 아이템을 개방하였다는 것을 검출하고, 협업 데이터와 같은 콘텐츠 아이템에 대한 추가 정보를 오버레이(overlay)에 제공할 수 있다. 예를 들어, 추가 정보는 콘텐츠 아이템에 대한 코멘트, 콘텐츠 아이템의 상태, 콘텐츠 아이템을 이전에 보았거나 또는 현재 보고 있는 다른 사용자의 활동을 포함할 수 있다. 이러한 오버레이는 다른 사용자가 현재 콘텐츠 아이템을 편집하고 있는 것으로 인해 변경 사항이 손실될 수 있다는 것을 사용자에게 경보할 수 있다.

[0072] 일부 실시형태에서, 위에서 논의된 서비스 또는 저장소/데이터베이스 중 하나 이상은 공개 또는 개인 애플리케이션 프로그래밍 인터페이스를 사용하여 액세스될 수 있다.

[0073] 특정 소프트웨어 애플리케이션은 사용자 대신 API를 통해 콘텐츠 저장소(142)에 액세스할 수 있다. 예를 들어, 클라이언트 디바이스(150)에서 실행되는 애플리케이션과 같은 소프트웨어 패키지는 사용자가 인증 자격 증명을 제공하고, 콘텐츠를 읽고, 기입하고, 생성하고, 삭제하거나 공유하거나 또는 조작할 때 콘텐츠 관리 시스템(110)에 API 호출을 프로그래밍 방식으로 직접 만들 수 있다.

[0074] 사용자는 웹 인터페이스 서비스(124)에 의해 생성되고 서빙되는 웹 인터페이스를 통해 사용자 계정에 저장된 콘

텐츠를 보거나 조작할 수 있다. 예를 들어, 사용자는 웹 브라우저에서 콘텐츠 관리 시스템(110)에 의해 제공되는 웹 주소로 내비게이션할 수 있다. 새로운 버전의 콘텐츠 아이템을 업로드하는 것과 같이 웹 인터페이스를 통해 만들어진 콘텐츠 저장소(142) 내의 콘텐츠에 대한 변경 또는 업데이트는 사용자의 계정과 연관된 다른 클라이언트 디바이스로 다시 전파될 수 있다. 예를 들어, 자체 클라이언트 소프트웨어를 각각 갖는 다수의 클라이언트 디바이스는 단일 계정과 연관될 수 있으며, 계정의 콘텐츠 아이템은 다수의 클라이언트 디바이스 각각 간에 동기화될 수 있다.

[0075] 클라이언트 디바이스(150)는 사용자를 대신하여 콘텐츠 관리 시스템(110)에 연결될 수 있다. 예를 들어, 클라이언트 디바이스(150)가 데스크탑 또는 랩탑 컴퓨터, 전화, 텔레비전, 사물 인터넷 디바이스 등인 경우, 사용자는 클라이언트 디바이스(150)와 직접 상호 작용할 수 있다. 대안적으로 또는 추가적으로, 클라이언트 디바이스(150)는 예를 들어, 클라이언트 디바이스(150)가 서버인 경우, 사용자가 클라이언트 디바이스(150)에 물리적으로 액세스하는 일 없이 사용자를 대신하여 행할 수 있다.

[0076] 클라이언트 디바이스(150)의 일부 특징은 클라이언트 디바이스(150)에 설치된 애플리케이션에 의해 구현된다. 일부 실시형태에서, 애플리케이션은 콘텐츠 관리 시스템별 구성 요소를 포함할 수 있다. 예를 들어, 콘텐츠 관리 시스템별 구성 요소는 독립형 애플리케이션(152), 하나 이상의 애플리케이션 플러그인 및/또는 브라우저 확장일 수 있다. 그러나, 사용자는 또한, 클라이언트 디바이스(150) 상에 상주하고 콘텐츠 관리 시스템(110)과 통신하도록 구성된 웹 브라우저와 같은 제3자 애플리케이션을 통해 콘텐츠 관리 시스템(110)과 상호 작용할 수 있다. 다양한 구현예에서, 클라이언트측 애플리케이션(152)은 사용자가 콘텐츠 관리 시스템(110)과 상호 작용하기 위한 사용자 인터페이스(UI)를 제공할 수 있다. 예를 들어, 사용자는 파일 시스템과 통합된 파일 시스템 탐색기 또는 웹 브라우저 애플리케이션을 사용하여 디스플레이되는 웹 페이지를 통해 콘텐츠 관리 시스템(110)과 상호 작용할 수 있다.

[0077] 일부 실시형태에서, 클라이언트 애플리케이션(152)은 콘텐츠 관리 시스템(110)의 하나 초과의 계정에 대한 콘텐츠를 관리 및 동기화하도록 구성될 수 있다. 이러한 실시형태에서, 클라이언트 애플리케이션(152)은 다수의 계정에 로그인된 상태로 유지되고 다수의 계정에 대한 정상적인 서비스를 제공할 수 있다. 일부 실시형태에서, 각각의 계정은 파일 시스템에서 폴더로서 나타날 수 있고, 이 폴더 내의 모든 콘텐츠 아이템은 콘텐츠 관리 시스템(110)과 동기화될 수 있다. 일부 실시형태에서, 클라이언트 애플리케이션(152)은 다수의 계정 중 하나를 기본 계정 또는 디폴트 계정이 되도록 선택하기 위한 선택기를 포함할 수 있다.

[0078] 콘텐츠 관리 시스템(110)에는 특정 구성 요소가 제공되지만, 이 기술 분야에 통상의 지식을 가진 자라면 시스템(100)의 아키텍처 구성은 단순히 하나의 가능한 구성일 뿐, 더 많거나 더 적은 구성 요소를 갖는 다른 구성도 가능하다는 것을 이해할 수 있을 것이다. 또한, 서비스는 심지어 다른 서비스에 있는 것으로 기술된 기능을 포함하여 더 많거나 더 적은 기능을 가질 수 있다. 또한, 실시형태와 관련하여 본 명세서에 설명된 특징은 다른 실시형태에 대해 설명된 특징과 결합될 수 있다.

[0079] 시스템(100)에는 특정 구성 요소가 제공되지만, 이 기술 분야에 통상의 지식을 가진 자라면 시스템(100)의 아키텍처 구성은 단순히 하나의 가능한 구성일 뿐, 더 많거나 더 적은 구성 요소를 갖는 다른 구성도 가능하다는 것을 이해할 수 있을 것이다.

[0080] 클라이언트 동기화 서비스

[0081] 도 2는 일부 실시형태에 따른 클라이언트 동기화 서비스(156)의 일례를 도시한다. 일부 실시형태에 따르면, 클라이언트 동기화 서비스(156)는 도 1의 클라이언트 디바이스에서 구현될 수 있다. 그러나, 다른 실시형태에서, 클라이언트 동기화 서비스(156)는 다른 컴퓨팅 디바이스에서 구현될 수 있다. 클라이언트 동기화 서비스(156)는 클라이언트 동기화 서비스(156)가 실행되는 클라이언트 디바이스와 콘텐츠 관리 시스템 간에 콘텐츠 아이템에 대한 변경 사항을 동기화하도록 구성된다.

[0082] 클라이언트 동기화 서비스(156)는 파일 시스템 인터페이스(205), 서버 인터페이스(210), 트리 저장소(220), 플래너(225) 및 스케줄러(230)를 포함할 수 있다. 추가적인 또는 대안적인 구성 요소도 또한 포함될 수 있다. 클라이언트 동기화 서비스(156) 및 그 구성 요소들에 대한 높은 레벨의 설명은 도 2와 관련하여 아래에서 논의된다. 그러나, 클라이언트 동기화 서비스(156) 및 그 구성 요소들에 대한 추가 상세 및 실시형태는 전체에 걸쳐 논의된다.

[0083] 파일 시스템 인터페이스(205)는 클라이언트 디바이스의 로컬 파일 시스템 상의 콘텐츠 아이템에 대한 변경 사항을 처리하고 로컬 트리를 업데이트하도록 구성된다. 예를 들어, 파일 시스템 인터페이스(205)는 도 1의 클라이

언트 동기화 서비스(156)와 통신하며, 클라이언트 디바이스의 로컬 파일 시스템 상의 콘텐츠 아이템의 변경 사항을 검출할 수 있다. 도 1의 클라이언트 애플리케이션(152)을 통해 변경이 이루어지고 검출될 수도 있다. 파일 시스템 인터페이스(205)는 클라이언트 디바이스 상의 콘텐츠 아이템에 대한 변경 사항(신규, 삭제, 수정, 복사, 이름 변경 또는 이동된 콘텐츠 아이템)에 기초하여 로컬 트리에 대한 업데이트가 이루어지게 할 수 있다.

[0084] 서버 인터페이스(210)는 콘텐츠 관리 시스템의 원격 저장소에서 콘텐츠 아이템에 대한 변경 사항을 원격으로 처리하고 원격 트리를 업데이트하는 것을 지원하도록 구성된다. 예를 들어, 서버 인터페이스(210)는 도 1의 서버 동기화 서비스(112)와 통신하며, 콘텐츠 관리 시스템(110)과 클라이언트 디바이스(150) 간에 콘텐츠 아이템에 대한 변경 사항을 동기화할 수 있다. 콘텐츠 관리 시스템에서 콘텐츠 아이템에 대한 변경 사항(신규, 삭제, 수정, 복사, 이름 변경 또는 이동된 콘텐츠 아이템)이 검출될 수 있고, 콘텐츠 관리 시스템(110)에서 변경 사항을 반영하기 위해 원격 트리에 업데이트가 이루어질 수 있다.

[0085] 트리 저장소(220)는 클라이언트 동기화 서비스(156)에 의해 사용된 트리 데이터 구조를 저장 및 유지하도록 구성된다. 예를 들어, 트리 저장소(220)는 로컬 트리, 동기 트리 및 원격 트리를 저장할 수 있다. 일부 실시형태에 따르면, 트리 저장소(200)는 대기 시간 및 응답 시간을 감소시키기 위해 트리 메모리 구조를 영구 메모리(예를 들어, 하드 디스크 또는 다른 2차 저장 디바이스)에 저장할 뿐만 아니라 주 메모리(예를 들어, RAM 또는 다른 1차 저장 디바이스)에 저장할 수 있다. 예를 들어, 클라이언트 디바이스 또는 클라이언트 동기화 서비스(156)의 시작 시에, 트리 데이터 구조는 영구 메모리로부터 검색되어 주 메모리로 로딩될 수 있다. 트리 저장소(220)는 주 메모리 상의 트리 데이터 구조에 액세스하고 이를 업데이트할 수 있고, 클라이언트 디바이스 또는 클라이언트 동기화 서비스(156)가 종료되기 전에 트리 저장소(220)는 업데이트된 트리 데이터 구조를 영구 메모리에 저장할 수 있다. 주 메모리는 비용이 많이 들고 대부분의 클라이언트 디바이스의 크기가 종종 제한되어 있기 때문에 주 메모리에서 트리 데이터 구조의 공간을 줄이기 위해 추가 기술적 개선이 구현된다. 이러한 기술적 해결책은 아래에 자세히 설명되어 있다.

[0086] 플래너(225)는 트리 데이터 구조의 상태에 기초하여 콘텐츠 관리 시스템과 연관된 서버 상태와 클라이언트 디바이스와 연관된 파일 시스템 상태 사이의 차이를 검출하도록 구성된다. 예를 들어, 플래너(225)는 원격 트리와 동기 트리 사이에 차이가 있는지를 결정할 수 있다. 원격 트리와 동기 트리 사이의 차이는 콘텐츠 관리 시스템에 저장된 하나 이상의 콘텐츠 아이템에 대해 원격으로 수행된 액션이 서버 상태와 파일 시스템 상태가 동기화되지 않게 하였다는 것을 나타낸다. 유사하게, 플래너(225)는 또한 로컬 트리와 동기 트리 사이에 차이가 있는지를 결정할 수 있다. 로컬 트리와 동기 트리 사이의 차이는 클라이언트 디바이스에 저장된 하나 이상의 콘텐츠 아이템에 대해 로컬로 수행된 액션이 서버 상태와 파일 시스템 상태가 동기화되지 않게 하였다는 것을 나타낸다. 차이가 검출되면, 플래너(225)는 트리 데이터 구조를 동기화하는 동작 세트를 생성한다.

[0087] 일부 시나리오에서, 원격 트리와 동기 트리 사이의 차이에 기초하여 생성된 동작 세트와, 로컬 트리와 동기 트리 사이의 차이에 기초하여 생성된 동작 세트가 충돌할 수 있다. 플래너(225)는 두 세트의 동작을 하나의 병합된 동작 계획으로 병합하도록 더 구성될 수 있다.

[0088] 스케줄러(230)는 생성된 동작 계획을 취하고 이 동작의 실행을 관리하도록 구성된다. 일부 실시형태에 따르면, 스케줄러(230)는 동작 계획의 각각의 동작을, 동작을 수행하기 위해 실행되어야 하는 하나 이상의 작업 시리즈로 변환한다. 일부 시나리오에서는 일부 작업이 오래되었거나 더 이상 관련이 없을 수 있다. 스케줄러(230)는 이러한 작업을 식별하고 작업을 취소하도록 구성된다.

[0089] 트리 데이터 구조

[0090] 도 3은 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한다. 트리 데이터 구조는 클라이언트 디바이스에 저장되고, 도 2의 클라이언트 동기화 서비스(156)와 같은 클라이언트 동기화 서비스에 의해 관리될 수 있다. 도 3에서, 트리 데이터 구조는 원격 트리(310), 동기 트리(330) 및 로컬 트리(350)를 포함하는 것으로 도시되어 있다.

[0091] 원격 트리(310)는 서버 상태를 나타내거나 또는 클라이언트 디바이스로부터 원격에 (예를 들어, 콘텐츠 관리 시스템의 서버에) 저장된 콘텐츠 아이템의 상태를 나타낸다. 로컬 트리(350)는 파일 시스템 상태를 나타내거나 또는 클라이언트 디바이스에 로컬로 저장된 대응하는 콘텐츠 아이템의 상태를 나타낸다. 동기 트리(330)는 로컬 트리와 원격 트리에 대한 병합 베이스를 나타낸다. 병합 베이스는 로컬 트리와 원격 트리의 공통 조상으로 고려되거나 또는 로컬 트리와 원격 트리 사이의 마지막으로 알려진 동기화 상태로 고려될 수 있다.

[0092] 각각의 트리 데이터 구조(예를 들어, 원격 트리(310), 동기 트리(330) 또는 로컬 트리(350))는 하나 이상의 노

드를 포함할 수 있다. 각 노드는 하나 이상의 자식 노드를 가질 수 있으며, 부모-자식 관계는 에지(edge)로 표시된다. 예를 들어, 원격 트리(310)는 노드(312 및 314)를 포함한다. 노드(312)는 노드(314)의 부모이고 노드(314)는 노드(312)의 자식이다. 이 부모-자식 관계는 에지(316)로 표현된다. 루트 노드(312)와 같은 루트 노드는 부모 노드를 갖지 않는다. 노드(314)와 같은 리프(leaf) 노드는 자식 노드를 갖지 않는다.

[0093] 트리 데이터 구조의 각 노드는 콘텐츠 아이템(예를 들어, 파일, 문서, 폴더 등)을 나타낼 수 있다. 예를 들어, 루트 노드(312)는 콘텐츠 관리 시스템과 연관된 루트 폴더를 나타낼 수 있고, 노드(314)는 이 루트 폴더에 위치된 파일(예를 들어, "Foo.txt"라는 텍스트 파일)을 나타낼 수 있다. 트리 데이터 구조의 각 노드는, 예를 들어, 콘텐츠 아이템의 부모 노드의 파일 식별자, 콘텐츠 아이템의 파일 이름, 콘텐츠 아이템에 대한 파일 식별자, 및 콘텐츠 아이템에 대한 메타데이터를 지정하는 디렉토리 파일 식별자("DirFileID")와 같은 데이터를 포함할 수 있다. 일부 실시형태에서, 트리 데이터 구조의 각 노드는 파일 식별자에 의해 키로 지시되거나 참조될 수 있고, 루트로부터 노드까지 고유 경로를 가질 수 있다.

[0094] 전술한 바와 같이, 클라이언트 동기화 서비스는 3개의 트리(예를 들어, 원격 트리(310), 동기 트리(330) 및 로컬 트리(350))가 모두 동일할 때 클라이언트 디바이스의 서버 상태와 파일 시스템 상태가 동기화된 것으로 결정할 수 있다. 다시 말해, 트리 구조와 이들이 나타내는 관계가 동일하고 노드에 포함된 데이터도 동일할 때 트리가 동기화된다. 역으로 3개의 트리가 동일하지 않으면 트리는 동기화되지 않는다. 도 3에 도시된 예시적인 시나리오에서, 원격 트리(310), 동기 트리(330) 및 로컬 트리(350)는 동일하고 동기화된 것으로 도시되고, 그 결과 서버 상태와 파일 시스템 상태가 동기화된다.

[0095] 트리 데이터 구조를 사용한 변경 사항 추적

[0096] 도 4는 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한다. 도 3에 도시된 트리 데이터 구조와 마찬가지로, 도 4에 도시된 트리 데이터 구조(원격 트리(410), 동기 트리(430) 및 로컬 트리(450)를 포함)는 클라이언트 디바이스에 저장될 수 있고, 도 2의 클라이언트 동기화 서비스(156)와 같은 클라이언트 동기화 서비스에 의해 관리될 수 있다. 도 3에는 트리 데이터 구조가 도시되어 있다.

[0097] 도 4는 도 3에 도시된 시나리오와 같이 이전에 동기화된 상태 후 트리에 표현된 콘텐츠 아이템에 대해 추가 액션이 수행되어 콘텐츠 아이템이 수정되어 더 이상 트리가 동기화되지 않은 시나리오를 도시한다. 동기 트리(430)는 이전에 알려진 동기화 상태의 표현을 유지하고, 서버 상태와 파일 시스템 상태 간의 차이를 식별하고 서버 상태와 파일 시스템 상태가 동기화되도록 수렴하도록 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스에 대한 동작을 생성하기 위해 클라이언트 동기화 서비스에 의해 사용될 수 있다.

[0098] 예를 들어, 사용자(클라이언트 디바이스와 연관된 사용자와 동일한 사용자 또는 콘텐츠 아이템에 액세스하는 다른 사용자)는 콘텐츠 관리 시스템에 의해 저장된 "foo.txt" 콘텐츠 아이템에 수정을 가할 수 있다. 이 콘텐츠 아이템은 원격 트리(410)에서 노드(414)로 표현된다. 원격 트리(410)에 도시된 수정은 foo.txt 콘텐츠 아이템의 제거(콘텐츠 관리 시스템에 의해 관리되는 공간으로부터 콘텐츠 아이템의 제거) 또는 삭제이다. 이러한 수정은 예를 들어 다른 클라이언트 디바이스에서 수행될 수 있고, 수정은 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 동기화되거나 또는 웹 브라우저를 통해 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 동기화되었다.

[0099] 콘텐츠 관리 시스템에서 변경이 이루어지면, 콘텐츠 관리 시스템은 변경이 이루어진 것을 지정하는 수정 데이터를 생성하고, 수정 데이터를 클라이언트 디바이스 상의 클라이언트 동기화 서비스로 전송한다. 클라이언트 동기화 서비스는 수정 데이터에 기초하여 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리를 업데이트한다. 예를 들어, 원격 트리(410)에서, foo.txt 콘텐츠 아이템을 나타내는 노드(414)는 삭제된 것으로 도시되어 있다.

[0100] 클라이언트 동기화 서비스는 원격 트리(410)와 동기 트리(430) 사이의 차이를 식별할 수 있고, 그 결과, 콘텐츠 관리 시스템에서 콘텐츠 아이템의 수정이 서버 상태와 파일 시스템 상태를 더 이상 동기화되지 않게 한 것으로 결정할 수 있다. 클라이언트 동기화 서비스는 서버 상태와 파일 시스템 상태를 수렴시켜 이들이 동기화되도록 구성된 클라이언트 디바이스 상에 저장된 콘텐츠 아이템에 대한 동작 세트 또는 동작 시퀀스를 더 생성 및 실행할 수 있다.

[0101] 추가적으로 또는 대안적으로, 사용자(콘텐츠 관리 시스템에서의 수정과 연관된 사용자와 동일한 사용자 또는 콘텐츠 아이템에 액세스하는 권한이 있는 다른 사용자)는 콘텐츠 관리 시스템과 연관된 클라이언트 디바이스 상에 로컬로 저장된 콘텐츠 아이템에 수정을 가할 수 있다. 예를 들어, 사용자는 "/bar" 폴더를 "/root" 폴더에 추가

하고 "Hi.doc" 문서를 "/bar" 폴더에 추가할 수 있다.

[0102] 클라이언트 디바이스에서 변경이 이루어질 때, 클라이언트 디바이스(예를 들어, 도 1의 클라이언트 동기화 서비스(156) 또는 클라이언트 애플리케이션(152))는 변경이 이루어진 것을 지정하는 수정 데이터를 생성하고, 클라이언트 디바이스의 클라이언트 동기화 서비스에 수정 데이터를 전달한다. 클라이언트 동기화 서비스는 수정 데이터에 기초하여 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대한 파일 시스템 상태를 나타내는 로컬 트리를 업데이트한다. 예를 들어, 로컬 트리(450)에서, 노드(452)와 노드(454)가 추가된 것으로 도시되어 있다. 노드(452)와 노드(454)는 각각 "/bar" 폴더와 "Hi.doc" 문서를 나타낸다.

[0103] 클라이언트 동기화 서비스는 로컬 트리(450)와 동기 트리(430) 사이의 차이를 식별할 수 있고, 그 결과 클라이언트 디바이스에서 콘텐츠 아이템의 수정이 서버 상태와 파일 시스템 상태가 더 이상 동기화되지 않게 한 것으로 결정할 수 있다. 클라이언트 동기화 서비스는 서버 상태와 파일 시스템 상태를 수렴시켜 이들이 동기화되도록 구성된 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대한 동작 세트 또는 동작 시퀀스를 더 생성할 수 있다. 이러한 동작은 실행을 위해 콘텐츠 관리 시스템으로 전송될 수 있다.

[0104] 도 4에 도시된 바와 같이, 클라이언트 디바이스에 저장된 콘텐츠 아이템과, 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대한 수정은 실질적으로 동시에 또는 특정 시간 기간 내에 일어날 수 있다. 이러한 수정은 트리 데이터 구조에 반영될 수 있고, 클라이언트 동기화 서비스에 의해 클라이언트 디바이스와 콘텐츠 관리 시스템에 대한 동작을 병렬로 생성하는데 사용될 수 있다. 그러나, 다른 시나리오에서, 동일한 시간 기간 내에 수정이 반드시 발생하는 것은 아닐 수 있으며 필요에 따라 동작이 생성될 수 있다. 또한, 도 4는 콘텐츠 아이템을 추가하고 콘텐츠 아이템을 삭제하기 위한 시나리오를 도시하지만, 콘텐츠 아이템의 편집, 이름 변경, 복사 또는 이동과 같은 다른 유형의 수정도 지원된다.

[0105] 다양한 실시형태에 따르면, 2개의 트리 데이터 구조 간의 차이를 식별하고 동작을 생성하는 것은 두 트리 데이터 구조에서 각각의 노드를 검사하고 노드에 대해 액션이 수행되었는지 여부를 결정하는 것을 포함할 수 있다. 액션은 예를 들어 노드의 추가, 노드의 삭제, 노드의 편집 또는 노드의 이동을 포함할 수 있다. 그런 다음 이 액션을 사용하여 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 동작을 생성할 수 있다.

[0106] 예를 들어, 2개의 트리 데이터 구조가 동기 트리와 원격 트리인 경우, 클라이언트 동기화 서비스는 예를 들어, 동기 트리 내의 모든 노드의 파일 식별자를 요청함으로써 동기 트리의 각 노드를 식별할 수 있다. 동기 트리에서 각 노드에 대해 또는 이 노드에 대한 파일 식별자에 대해, 클라이언트 동기화 서비스는 노드 또는 파일 식별자가 또한 원격 트리에 있는지 여부를 결정할 수 있다. 원격 트리에서 발견되지 않는 동기 트리의 노드 또는 파일 식별자는 원격 트리가 나타내는 서버 상태로부터 노드가 삭제되었음을 나타낼 수 있다. 따라서, 클라이언트 동기화 서비스는 원격 트리에서 삭제 액션이 발생한 것으로 결정할 수 있다. 노드 또는 이 노드의 파일 식별자가 원격 트리에서 발견되면, 클라이언트 동기화 서비스는 원격 트리의 노드가 편집 또는 이동되었는지 여부를 검사할 수 있다.

[0107] 원격 트리의 노드가 동기 트리의 노드에 대해 편집되었는지 여부를 결정하기 위해, 클라이언트 동기화 서비스는 동기 트리의 노드에 대한 메타데이터를 원격 트리의 대응하는 노드(예를 들어, 동일한 파일 식별자를 갖는 노드)에 대한 메타데이터와 비교할 수 있다. 메타데이터는 노드에 의해 표현된 콘텐츠 아이템이 편집되었는지 여부를 결정하기 위해 사용될 수 있는 정보를 포함할 수 있다. 예를 들어, 메타데이터는 콘텐츠 아이템 또는 그 일부의 데이터에 기초하여 생성된 하나 이상의 해시 값을 포함할 수 있다. 메타데이터는 추가적으로 또는 대안적으로 콘텐츠 아이템에 대한 크기 값, 최종 수정된 값 또는 다른 값을 포함할 수 있다. 동기 트리에서 노드에 대한 메타데이터는 원격 트리에서 노드에 대한 메타데이터와 비교될 수 있다. 메타데이터가 일치하지 않으면 콘텐츠 아이템의 편집이 원격 트리로 표시되는 서버 상태에서 편집되었을 수 있다. 따라서, 클라이언트 동기화 서비스는 원격 트리 상의 노드에 대해 편집 액션이 발생했다고 결정할 수 있다. 메타데이터가 일치하면 편집이 수행되지 않았을 수 있다.

[0108] 원격 트리 내의 노드가 이동되었는지를 결정하기 위해, 클라이언트 동기화 서비스는 동기 트리 내의 노드의 위치를 원격 트리 내의 대응하는 노드(예를 들어, 동일한 파일 식별자를 갖는 노드)의 위치와 비교할 수 있다. 이 위치는 예를 들어 노드가 위치된 경로, 파일 이름, 및/또는 노드 부모의 파일 식별자를 지정하는 디렉토리 파일 식별자("DirFileID")를 포함할 수 있다. 위치가 일치하면 이동이 발생하지 않았을 수 있다. 한편, 위치가 일치하지 않으면, 원격 트리가 나타내는 서버 상태에서 콘텐츠 아이템의 이동이 발생했을 수 있다. 따라서, 클라이언트 동기화 서비스는 원격 트리 상의 노드에 대해 이동 액션이 발생했다고 결정할 수 있다.

[0109] 노드가 원격 트리에 추가되었는지를 결정하기 위해, 클라이언트 동기화 서비스는 동기 트리에서 발견되지 않은 원격 트리 내 임의의 노드 또는 파일 식별자를 식별할 수 있다. 노드 또는 파일 식별자가 원격 트리에서 발견되고 동기 트리에서 발견되지 않으면, 클라이언트 동기화 서비스는 이 노드의 추가 액션이 서버 상태를 나타내는 원격 트리에서 발생한 것으로 결정할 수 있다.

[0110] 상기 예가 동기 트리와 원격 트리에 대해 설명되었지만, 다른 실시형태에서, 동기 트리와 로컬 트리 사이의 차이를 식별하고 파일 시스템 상태를 나타내는 로컬 트리에서 발생한 액션이 무엇인지 결정하기 위해 동기 트리 및 로컬 트리와 유사한 프로세스가 발생할 수 있다.

[0111] 트리 데이터 구조를 사용한 동기화

[0112] 도 5는 본 기술의 다양한 실시형태에 따라 트리 데이터 구조를 사용하여 서버 상태와 파일 시스템 상태를 동기화하기 위한 예시적인 방법을 도시한다. 본 명세서에 기술된 방법 및 프로세스는 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(500)은 클라이언트 디바이스에서 실행되는 도 2의 예를 들어 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0113] 시스템은 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리, 클라이언트 디바이스에 저장된 대응하는 콘텐츠 아이템에 대한 파일 시스템 상태를 나타내는 로컬 트리, 및 서버 상태와 파일 시스템 상태 간의 알려진 동기화 상태를 나타내는 동기 트리 사이의 차이를 식별하도록 구성된다. 이들 차이에 기초하여, 실행되면, 서버 상태와 파일 시스템 상태를 3개의 트리 데이터 구조를 동일하게 하는 동기화 상태로 수렴하도록 구성된 동작 세트가 생성될 수 있다.

[0114] 예를 들어, 동작(505)에서, 시스템은 콘텐츠 관리 시스템에 의해 또는 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대한 수정 데이터를 수신할 수 있다. 수정 데이터는 동작(510)에서 원격 트리 또는 로컬 트리를 업데이트하기 위해 사용될 수 있다.

[0115] 수정 데이터는 콘텐츠 관리 서비스와 연관된 하나 이상의 콘텐츠 아이템에 대해 변경 사항이 무엇인지를 지정한다. 따라서, 수정 데이터는 콘텐츠 관리 시스템 또는 클라이언트 디바이스(예를 들어, 도 1의 클라이언트 디바이스(150)에서 실행되는 클라이언트 애플리케이션(152))로부터 수신될 수 있다. 콘텐츠 관리 시스템으로부터 수신된 수정 데이터는 서버 수정 데이터로 지칭될 수 있다. 서버 수정 데이터는 콘텐츠 관리 시스템에 의해 하나 이상의 콘텐츠 아이템에 수행된 변경 사항이 무엇인지를 지정하고, 동작(510)에서 원격 트리를 업데이트하는데 사용될 수 있다. 클라이언트 디바이스로부터 수신된 수정 데이터는 클라이언트 수정 데이터로 지칭될 수 있다. 클라이언트 수정 데이터는 클라이언트 디바이스 상의 하나 이상의 콘텐츠 아이템에 이루어진 변경 사항이 무엇인지를 지정하고, 동작(510)에서 로컬 트리를 업데이트하는 데 사용될 수 있다.

[0116] 동작(515)에서, 시스템은 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대한 서버 상태와 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대한 파일 시스템 상태가 동기화되어 있는지 여부를 결정할 수 있다. 로컬 트리와 원격 트리는 파일 시스템 상태와 서버 상태를 나타내며 콘텐츠 관리 시스템과 클라이언트 디바이스에서 발생하는 변경 사항을 추적하기 위해 지속적으로 업데이트되기 때문에 서버 상태와 파일 시스템 상태가 동기화되어 있는지 여부를 결정하는 것은 로컬 트리 및/또는 원격 트리를 동기 트리와 비교하여 트리들 사이의 차이를 찾아냄으로써 수행될 수 있다. 트리들 사이의 차이를 발견하는 이 과정은 때때로 트리를 "디핑(diffing)"하는 것이라고 지칭된다.

[0117] 일부 실시형태 및 시나리오에 따르면, 서버 상태와 파일 시스템 상태가 동기화되는지 여부를 결정하는 것은 원격 트리와 동기 트리 사이의 차이를 식별하는 것 및/또는 로컬 트리와 동기 트리 사이의 차이를 식별하는 것 중 하나 이상을 포함할 수 있다. 원격 트리와 동기 트리 사이의 차이는 클라이언트 디바이스에 반영되지 않았을 수 있는 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 변경 사항이 발생한 것을 나타낼 수 있다. 유사하게, 로컬 트리와 동기 트리 사이의 차이는 콘텐츠 관리 시스템에 반영되지 않았을 수 있는 클라이언트 디바이스에 저장된 콘텐츠 아이템에 변경 사항이 발생한 것을 나타낼 수 있다.

[0118] 트리들 간에 차이가 없다면, 서버 상태와 파일 시스템 상태는 동기화되어 있어서 동기화 액션이 필요하지 않다. 따라서, 방법은 동작(505)으로 되돌아가서 새로운 수정 데이터를 기다릴 수 있다. 한편, 차이가 검출되면, 시스템은 동작(520)에서 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 동작 세트를 생성할 수 있다.

[0119] 생성된 동작 세트는 검출된 하나 이상의 차이에 의존한다. 예를 들어, 두 트리 간의 차이가 추가된 콘텐츠 아이

템인 경우, 생성된 동작 세트는 추가된 콘텐츠 아이템을 검색하고 이를 추가하는 것을 포함할 수 있다. 두 트리 간의 차이가 콘텐츠 아이템의 삭제인 경우, 생성된 동작 세트는 콘텐츠 아이템을 삭제하는 것을 포함할 수 있다. 일부 실시형태에 따르면, 동작 세트는 또한 트리 제약이 유지되는 것을 보장하기 위해 다수의 검사를 포함할 수 있다. 아래에 더 설명되는 바와 같이, 동작 세트는 서버 상태, 파일 시스템 상태, 또는 실행 보류 중인 다른 동작의 현재 상태와 충돌할 수 있다. 따라서 시스템은 진행하기 전에 이러한 충돌을 해결할 수도 있다.

[0120] 전술한 바와 같이, 원격 트리와 동기 트리 사이에 차이가 있는 경우, 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에서 클라이언트 디바이스에 반영되지 않았을 수 있는 변경 사항이 일어났을 수 있다. 따라서, 이 시나리오에서, 시스템은 서버 상태와 파일 시스템 상태를 수렴하기 위해 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대해 동작하도록 구성된 클라이언트 동작 세트를 생성할 수 있고, 이 클라이언트 동작 세트는 동작(525)에서 실행을 위해 클라이언트 디바이스에 제공될 수 있다.

[0121] 한편, 로컬 트리와 동기 트리 사이에 차이가 있는 경우, 클라이언트 디바이스에 저장된 콘텐츠 아이템에서 콘텐츠 관리 시스템에 반영되지 않았을 수 있는 변경 사항이 발생했을 수 있다. 따라서, 이 시나리오에서, 시스템은 서버 상태와 파일 시스템 상태를 수렴하기 위해 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대해 동작하도록 구성된 서버 동작 세트를 생성할 수 있고, 이 서버 동작 세트는 동작(525)에서 실행을 위해 콘텐츠 관리 시스템에 제공될 수 있다. 일부 경우에, 두 경우는 모두 참일 수 있고, 클라이언트 동작 세트와 서버 동작 세트가 생성되어 동작(525)에서 의도된 수신자에게 제공될 수 있다.

[0122] 일단 동작 세트(들)가 의도된 수신자(들)에게 제공되면, 방법은 동작(505)으로 되돌아가서 새로운 수정 데이터를 기다릴 수 있다. 동작 세트(들)는 서버 상태와 파일 시스템 상태를 수렴시키는 쪽으로 하나 이상의 단계를 제공할 수 있고, 또는 서버 상태와 파일 시스템 상태를 동기화하는데 필요한 모든 단계를 제공할 수 있다. 예를 들어, 콘텐츠 관리 시스템은 서버 동작 세트를 수신하고 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대해 서버 동작 세트를 실행할 수 있다. 서버 동작 세트를 실행하면 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 변경을 야기하며, 이 변경은 서버 수정 데이터에서 검출 및 지정되고, 시스템으로 다시 전달된다. 시스템은 원격 트리를 업데이트하고 서버 상태와 파일 시스템 상태가 동기화되어 있는지 여부를 결정할 수 있다.

[0123] 클라이언트 디바이스는 클라이언트 동작 세트를 수신하고, 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대해 클라이언트 동작 세트를 실행할 수 있다. 클라이언트 동작 세트를 실행하면 클라이언트 디바이스에 저장된 콘텐츠 아이템에 변경을 야기하며, 이 변경은 클라이언트 수정 데이터에서 검출 및 지정되고, 시스템으로 전달된다. 시스템은 로컬 트리를 업데이트하고 서버 상태와 파일 시스템 상태가 동기화되어 있는지 여부를 결정할 수 있다. 방법(500)의 이러한 동작은 서버 상태와 파일 시스템 상태가 동기화될 때까지 계속될 수 있다.

[0124] 방법(500)의 동작은 클라이언트 측과 서버 측(예를 들어, 로컬 트리와 원격 트리, 파일 시스템 상태와 서버 상태, 클라이언트 동작 세트와 서버 동작 세트, 클라이언트 수정 데이터와 서버 수정 데이터)에 대해 설명된다. 다양한 실시형태에서, 2개의 측과 연관된 동작은 다른 측과 병렬로, 순차적으로, 별도로, 또는 조합으로 일어날 수 있다.

[0125] 더 상세히 논의될 바와 같이, 일부 실시형태에 따르면, 동작이 실행을 위해 제공되기 전에, 시스템은 동작이 규칙 세트 또는 불변 사항 세트를 따르는지 여부를 결정하기 위해 동작을 검사할 수 있다. 동작이 규칙을 위반하면 시스템은 규칙 위반과 연관된 해결 프로세스를 실행한다.

[0126] 추가적으로, 일부 실시형태에 따르면, 시스템(예를 들어, 도 2의 클라이언트 동기화 서비스(156)의 스케줄러(230))은 동작 세트의 실행을 관리할 수 있다. 예를 들어, 동작 세트 내의 각 동작은 작업, 실행 스레드, 단계 시리즈 또는 명령과 연관될 수 있다. 시스템은 작업, 스레드, 단계 또는 명령을 실행하고 클라이언트 디바이스 및/또는 콘텐츠 관리 시스템과 인터페이스하며 동작 세트를 실행하고 서버 상태와 파일 시스템 상태를 수렴시키도록 구성될 수 있다.

[0127] 충돌 처리

[0128] 도 5와 관련하여 전술한 바와 같이, 동기 트리와 원격 트리 사이의 차이가 식별되어 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 클라이언트 동작 세트를 생성하는데 사용된다. 그러나 경우에 따라 클라이언트 동작 세트는 로컬 트리의 현재 상태와 충돌할 수 있다. 유사하게, 동기 트리와 로컬 트리 간의 차이가 식별되어 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 서버 동작 세트를 생성하는 데 사용된다. 그러나 서버 동작 세트는 원격 트리의 현재 상태와 충돌할 수 있다. 추가적으로 또는 대안적으로, 클라이언트 동작 세트 및 서버 동작 세트는 서로 충돌하거나, 시스템에 의해 유지되는 다른 규칙 또는 불변 사항을 위반할 수 있다. 따라서, 본 기

술의 다양한 실시형태는 이러한 충돌을 해결함으로써 추가적인 기술적 개선을 제공한다.

[0129] 예를 들어, 도 2의 클라이언트 동기화 서비스(156) 내의 플래너(225)는 규칙과 충돌하는 동작 세트(예를 들어, 클라이언트 동작 세트 또는 서버 동작 세트)의 동작을 식별할 수 있다. 충돌을 식별하는 데 사용되는 각 규칙은 충돌 해결과 연관될 수도 있다. 클라이언트 동기화 서비스는 충돌 해결에 기초하여 동작 세트를 업데이트할 수 있고, 또는 실행 동작 세트를 제공하기 전에 충돌 해결과 연관된 동작을 수행함으로써 충돌 해결을 수행할 수 있다.

[0130] 도 6은 본 기술의 다양한 실시형태에 따라 트리 데이터 구조를 사용하여 서버 상태와 파일 시스템 상태를 동기화할 때 충돌을 해결하기 위한 예시적인 방법(600)을 도시한다. 본 명세서에 기술된 방법 및 프로세스가 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(600)은 예를 들어 클라이언트 디바이스에서 실행되는 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0131] 시스템은 동작(620)에서 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 동작 세트를 수신할 수 있다. 동작 세트는 예를 들어 도 5의 방법(500)과 관련하여 생성되고 기술된 클라이언트 동작 세트, 서버 동작 세트, 또는 조합된 동작 세트일 수 있다.

[0132] 동작(650)에서, 시스템은 규칙 세트에 기초하여 동작 세트에서 하나 이상의 위반을 식별한다. 규칙 세트는 도 2의 클라이언트 동기화 서비스(156)에 의해 저장될 수 있고, 동작에 대해 해결해야 할 여러 제약, 불변 사항 또는 충돌을 지정할 수 있다. 규칙 세트는 트리 데이터 구조에 적용될 수 있으며, 동기화 거동을 제어하는 데 도움이 된다. 규칙 세트의 각 규칙은 또한 이 규칙 위반에 대한 해결과 연관되거나 연결될 수 있다. 예를 들어, 해결은 동작 세트에서 하나 이상의 동작의 변경, 하나 이상의 동작의 제거, 하나 이상의 동작의 추가, 서버 상태 또는 파일 상태에 하나 이상의 추가 액션, 또는 액션의 조합을 포함할 수 있다.

[0133] 동작 세트의 각 동작에 대해, 시스템은 규칙 세트 내의 임의의 규칙이 위반되는지 여부를 결정할 수 있다. 규칙이 위반되면, 시스템은 위반의 해결을 식별하고, 동작(655)에서, 해결을 수행한다. 해결은 동작 세트에서 하나 이상의 동작 수정, 하나 이상의 동작 제거 또는 추가, 또는 서버 상태 또는 파일 상태에 대한 추가 액션과 같은 액션을 포함할 수 있다.

[0134] 해결 액션이 수행되면, 시스템은 동작(660)에서 해결 및 동작 세트에 기초하여 해결되거나 재 기반된 동작 세트를 생성할 수 있고, 동작(665)에서, 해결된 동작 세트를 적절한 실행 엔티티에 제공할 수 있다. 예를 들어, 해결된 동작 세트는 관리되는 실행을 위해 도 2의 클라이언트 동기화 서비스(146)의 스케줄러(230)에 제공될 수 있다. 대안적으로, 동작 세트가 클라이언트 동작 세트이면, 해결된 동작 세트는 클라이언트 디바이스에 제공될 수 있다. 동작 세트가 서버 동작 세트이면, 해결된 동작 세트는 콘텐츠 관리 서비스에 제공될 수 있다. 추가적으로, 도 6의 방법(600)은 클라이언트 동작 세트와 서버 동작 세트에 순차적으로, 병렬로 또는 다양한 상이한 순서로 수행될 수 있다.

[0135] 일부 실시형태에 따르면, 각 동작 유형은 동일하거나 상이한 규칙 세트와 연관될 수 있다. 예를 들어, 동작 유형은 예를 들어 콘텐츠 아이템 추가, 콘텐츠 아이템 삭제, 콘텐츠 아이템 편집, 콘텐츠 아이템 이동, 콘텐츠 아이템 이름 변경 등을 포함할 수 있다. 동작 세트는 위의 동작 유형 중 하나에 각각 속하는 동작으로 구성될 수 있다. 각 동작 유형은 특정 규칙 세트와 연관될 수 있다.

[0136] 예시적인 목적으로, "추가" 동작 유형에 대한 규칙 세트는 콘텐츠 아이템에 대한 파일 식별자가 트리에서 고유해야 한다(예를 들어, 트리에서 2개의 노드가 동일한 파일 식별자를 가질 수 없다)는 규칙, 콘텐츠 아이템의 부모 노드의 파일 식별자를 지정하는 디렉토리 파일 식별자("DirFileID")는 반대 트리 데이터 구조에 존재해야 한다는 규칙, 및 콘텐츠 아이템에 대한 DirFileID와 파일 이름 조합은 반대 트리에서 사용되지 않는다는 규칙을 포함할 수 있다.

[0137] 여기에 사용된 반대 트리는 반대 엔티티의 상태를 나타내는 트리 데이터 구조를 지칭한다. 예를 들어 클라이언트 디바이스에서 동작하도록 구성된 클라이언트 동작 세트와, 클라이언트 디바이스의 파일 시스템에 생성된 변경 사항은 로컬 트리에 반영된다. 따라서 클라이언트 동작 세트의 반대 트리는 원격 트리이다. 유사하게, 서버 동작 세트는 실행될 콘텐츠 관리 시스템으로 전송되도록 구성되고, 서버 상태에 생성된 변경 사항은 원격 트리에 반영된다. 따라서 서버 동작 세트의 반대 트리는 로컬 트리이다.

[0138] 도 7은 다양한 실시형태에 따른, 추가 동작에 대한 규칙 위반을 나타내는 트리 데이터 구조의 일례를 도시한다.

트리 데이터 구조는 원격 트리(710), 동기 트리(750) 및 로컬 트리(770)를 포함한다. 로컬 트리(770)를 참조할 때, 원격 트리(710)는 반대 트리로 고려될 수 있다. 한편, 원격 트리(710)를 참조할 때, 로컬 트리(770)는 반대 트리로 고려될 수 있다. 도 7은 원격 트리(710)에서 노드(712)에 의해 표현된 콘텐츠 아이템을 추가하는 동작 세트를 도시한다. 예를 들어, 클라이언트 동기화 서비스는 원격 트리(710)를 동기 트리(750)와 비교하고, 차이를 식별하고, 노드(712)의 추가를 포함하는 동작 세트를 생성할 수 있다. 노드(712)는 4의 FileID, 3의 DirFileID(노드(712)의 부모인 부모 노드(714)를 참조함), 및 "Hi"의 파일 이름과 연관된다. 부모 노드(714)는 3의 FileID, 1의 DirFileID(노드(714)의 부모인 루트 노드(716)를 참조함), 및 "Foo"의 파일 이름과 연결된다.

[0139] 클라이언트 동기화 서비스는 도 6의 방법(600)을 수행할 수 있고, 노드(712)에 대한 추가 동작이 "추가" 동작 유형에 대해 "콘텐츠 아이템의 디렉토리 파일 식별자("DirFileID")가 반대 트리 데이트 구조에 존재해야 한다"는 규칙을 위반하고 있는 것으로 결정할 수 있다. 이것은 도 7에 로컬 트리(770)가 노드(712)의 부모 노드(714)를 참조하는 3의 file ID를 갖는 노드를 갖지 않는 것으로 도시되어 있다. 이것은 예를 들어, 원격 트리(710)와 동기 트리(750) 사이의 차이가 결정되고 동작 세트가 생성된 후, 노드(714)에 대응하는 "Foo" 노드가 반대 트리로부터 제거될 때 일어날 수 있다.

[0140] 이 규칙과 연관된 해결은 로컬 트리(770)로부터 누락된 노드를 동기 트리(750)로부터 삭제하여 동기 트리(750)와 로컬 트리(770)를 동기화하고 원격 트리(710)와 동기 트리(750)를 재 디핑(rediffing)하는 (예를 들어, 원격 트리와 동기 트리 사이의 차이를 찾는) 것을 포함할 수 있다. 도 7에 도시된 시나리오에서 동기 트리(750)의 노드(754)는 제거되고(758), 원격 트리(710)와 동기 트리(750) 사이의 차이를 식별하기 위해 디핑 동작이 시작된다. 이것은 동작 세트에 노드(714)의 추가 동작과, 노드(712)의 추가 동작을 포함하게 한다.

[0141] 유사하게, "추가" 동작 유형에 대해 "콘텐츠 아이템에 대한 파일 식별자는 트리에서 고유해야 한다는" 규칙의 위반은 콘텐츠 관리 시스템으로부터 추가된 노드에 대한 새로운 파일 ID를 요청하고, 노드를 추가할 때 새로운 파일 ID를 사용하는 것을 포함하는 동작에 의해 해결될 수 있다. "추가" 동작 유형에 대해 "콘텐츠 아이템에 대해 DirFileID와 파일 이름 조합이 반대 트리에서 사용되지 않는다는" 규칙의 위반은 두 노드와 연관된 메타데이터를 통해 콘텐츠 아이템이 동일한지 여부를 검사하는 것을 포함하는 동작에 의해 해결될 수 있다. 콘텐츠 아이템이 동일한 경우 추가되는 콘텐츠 아이템이 이미 다른 액션에서 추가되었을 수 있다. 콘텐츠 아이템이 동일하지 않은 경우 추가되는 콘텐츠 아이템의 파일 이름이 변경되었을 수 있다. 예를 들어, 추가되는 콘텐츠 아이템의 파일 이름에 "(충돌된 버전)"이라는 텍스트가 추가될 수 있다.

[0142] 증분적인 플래너

[0143] 도 3, 도 4, 및 도 7에 도시된 다양한 트리 데이터 구조는 비교적 적은 수의 노드를 포함하고 비교적 단순한 구조이지만, 시스템에 의해 지원되는 트리 데이터 구조는 다수의 레벨 및 각 레벨에서 잠재적으로 다수의 노드를 가져서 훨씬 더 크고 복잡할 수 있다. 따라서, 동작 동안 트리 데이터 구조를 저장하는데 필요한 메모리 사용은 상당히 클 수 있고, 트리 데이터 구조에서 동작하는데 필요한 컴퓨팅 시간 및 자원은 상당히 클 수 있다. 예를 들어, 원격 트리와 동기 트리 간의 차이 및/또는 로컬 트리와 동기 트리 간의 차이를 찾고, 원격 트리와 동기 트리 및/또는 로컬 트리와 동기 트리를 수렴하는 데 필요한 동작을 생성하려면 많은 양의 메모리, 시간 및 다른 컴퓨팅 자원을 요구할 수 있다.

[0144] 불행하게도, 이들 컴퓨팅 자원은 제한적이다. 예를 들어, 클라이언트 디바이스는 이용 가능한 메모리의 양이 제한되어 있고, 트리를 디핑하고 동작을 생성하는 데 필요한 시간 기간은 클라이언트 디바이스, 클라이언트 애플리케이션, 또는 콘텐츠 관리 시스템에서 제공하는 콘텐츠 관리 서비스의 사용 가능성을 방해할 수 있다. 또한 서버 상태와 파일 시스템 상태를 수렴하는 데 필요한 시간이 많을수록 두 상태에 개재하는 변경이 동작 세트를 계산 또는 실행시키고 및/또는 타깃 동기화 상태가 오래된 것일 가능성이 높아진다. 따라서, 본 기술의 다양한 실시형태는 서버 상태와 파일 시스템 상태를 나타내는 트리 데이터 구조와 함께 서버 상태와 파일 시스템 상태를 증분적으로 수렴함으로써 추가적인 기술적 개선을 제공한다.

[0145] 도 8은 본 기술의 다양한 실시형태에 따른, 서버 상태와 파일 시스템 상태를 증분적인 수렴하기 위한 예시적인 방법(800)을 도시한다. 본 명세서에 기술된 방법 및 프로세스가 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(800)은 예를 들어 클라이언트 디바이스에서 실행되는 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0146] 동작(805)에서, 시스템은 원격 트리 또는 로컬 트리를 업데이트하는데 사용될 수 있는 수정 데이터를 수신할 수

있다. 예를 들어, 서버 수정 데이터는 콘텐츠 관리 서비스로부터 수신될 수 있고, 콘텐츠 관리 시스템에 의해 저장된 하나 이상의 콘텐츠 아이템과 연관된 수정 또는 다른 액션(예를 들어, 편집, 추가, 삭제, 이동 또는 이름 변경)을 지정할 수 있다. 서버 수정 데이터는 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템의 서버 상태를 나타내는 원격 트리를 업데이트하는데 사용될 수 있다. 유사하게, 클라이언트 수정 데이터는 클라이언트 디바이스(예를 들어, 클라이언트 애플리케이션)로부터 수신될 수 있고, 클라이언트 디바이스에 저장된 하나 이상의 콘텐츠 아이템과 연관된 수정 또는 다른 액션을 지정할 수 있다. 클라이언트 수정 데이터는 클라이언트 디바이스에 저장된 콘텐츠 아이템의 파일 시스템 상태를 나타내는 로컬 트리를 업데이트하는데 사용될 수 있다.

[0147] 콘텐츠 아이템과 연관된 수정을 지정하는 수신된 수정 데이터에 기초하여, 시스템은 동작(810)에서 수정된 콘텐츠 아이템에 대응하는 노드를 식별하고, 노드를 수정된 콘텐츠 아이템의 리스트에 추가할 수 있다(예를 들어, 노드와 연관된 파일 식별자를 수정된 콘텐츠 아이템의 리스트에 추가할 수 있다). 동작(805 및 810)은 시스템이 방법(800)의 다음 단계로 진행하기 전에 일정 시간 동안 연속적으로 발생할 수 있다. 예를 들어, 추가 수정 데이터가 수신되고, 시스템에 의해 관리되는 트리를 업데이트하고 노드를 수정된 콘텐츠 아이템의 리스트에 추가하는데 사용될 수 있다.

[0148] 서버 상태와 파일 시스템 상태를 증분적으로 수렴하기 위해, 시스템은 동작(815)에서 수정된 콘텐츠 아이템의 리스트에서 각 노드를 취하고, 노드가 어떻게 수정되었는지(예를 들어, 어떤 액션이 노드와 연관되는지)를 결정한다. 일부 실시형태에서, 수정 데이터는 수정을 노드에 지정할 수 있다. 그러나, 다른 실시형태에서, 시스템은 원격 트리와 동기 트리의 비교 및/또는 로컬 트리와 동기 트리의 비교에 기초하여 노드에 대한 수정 사항을 결정할 수 있다. 예를 들어, 수정은 노드의 추가, 노드의 삭제, 노드의 편집 또는 노드의 이동을 포함할 수 있다.

[0149] 수정된 콘텐츠 아이템의 리스트에서 각각의 노드 또는 노드에 대한 파일 식별자에 대해, 시스템은, 만약 있다면, 노드에서 수행된 수정이 무엇인지를 결정하기 위해 일련의 검사를 수행할 수 있다. 예를 들어, 시스템은 파일 식별자가 동기 트리에는 있지만 원격 트리에는 없는지 여부를 결정할 수 있다. 원격 트리에서 발견되지 않는 동기 트리의 파일 식별자는 원격 트리로 표시되는 서버 상태로부터 노드가 삭제되었음을 나타낼 수 있다. 따라서, 클라이언트 동기화 서비스는 노드의 삭제 수정이 원격 트리에서 발생하였다고 결정할 수 있다. 유사하게, 시스템은 또한 파일 식별자가 동기 트리에는 있지만 로컬 트리에는 없는지 여부를 결정할 수 있다. 로컬 트리에서 발견되지 않는 동기 트리의 파일 식별자는 노드가 로컬 트리로 표시되는 파일 시스템 상태로부터 삭제되었음을 나타낼 수 있다. 따라서, 클라이언트 동기화 서비스는 노드의 삭제 수정이 로컬 트리에서 발생하였다고 결정할 수 있다.

[0150] 노드에 편집 수정이 수행되었는지 여부를 결정하기 위해, 시스템은 동기 트리에서 노드에 대한 메타데이터를 원격 트리 및/또는 로컬 트리에서 대응하는 노드(예를 들어, 동일한 파일 식별자를 갖는 노드)에 대한 메타데이터와 비교할 수 있다. 메타데이터는 노드로 표현된 콘텐츠 아이템이 편집되었는지 여부를 결정하기 위해 사용될 수 있는 정보를 포함할 수 있다. 예를 들어, 메타데이터는 콘텐츠 아이템 또는 그 일부의 데이터에 기초하여 생성된 하나 이상의 해시 값을 포함할 수 있다. 메타데이터는 콘텐츠 아이템에 대한 크기 값, 최종 수정된 값 또는 다른 값을 추가적으로 또는 대안적으로 포함할 수 있다. 메타데이터가 일치하지 않은 경우, 원격 트리로 표현된 서버 상태 및/또는 로컬 트리로 표현된 파일 시스템 상태에서 콘텐츠 아이템의 편집이 편집되었을 수 있다. 따라서, 시스템은 원격 트리 및/또는 로컬 트리의 노드에 편집 액션이 발생하였다고 결정할 수 있다.

[0151] 원격 트리의 노드가 이동되었는지 여부를 결정하기 위해, 시스템은 동기 트리의 노드의 위치를 원격 트리 및/또는 로컬 트리의 대응하는 노드(예를 들어, 동일한 파일 식별자를 갖는 노드)의 위치와 비교할 수 있다. 위치는 예를 들어 노드가 위치된 경로, 파일 이름, 및/또는 노드 부모의 파일 식별자를 지정하는 디렉토리 파일 식별자("DirFileID")를 포함할 수 있다. 위치가 일치하면 이동이 발생하지 않았을 수 있다. 한편, 위치가 일치하지 않으면 원격 트리 또는 로컬 트리에서 콘텐츠 아이템의 이동이 일어났을 수 있다. 따라서, 클라이언트 동기화 서비스는 원격 트리 및/또는 로컬 트리의 노드에 이동 액션이 일어났다고 결정할 수 있다.

[0152] 노드가 원격 트리에 추가되었는지 여부를 결정하기 위해, 시스템은 수정된 콘텐츠 아이템의 리스트의 파일 식별자가 원격 트리 또는 로컬 트리에는 있지만 동기 트리에는 없는지 여부를 결정할 수 있다. 파일 식별자가 원격 트리 또는 로컬 트리에서는 발견되고 동기 트리에서는 발견되지 않으면, 시스템은 이 노드에 추가 수정이 발생했다고 결정할 수 있다.

[0153] 수정된 콘텐츠 아이템의 리스트의 노드에 하나 이상의 수정이 있다고 결정되면, 시스템은 동작(820)에서 이들 수정 중 임의의 수정이 의존성을 갖는지 여부를 결정할 수 있다. 도 9와 관련하여 더 예시된 바와 같이, 예를 들어, 다른 수정이 먼저 발생하지 않고는 수정이 실행될 수 없을 때 노드의 수정은 의존성을 갖는다.

[0154] 수정이 의존성을 갖지 않으면, 시스템은 동작(825)에서 차단되지 않은 액션 리스트에 수정을 추가한다. 수정이 의존성을 갖는 경우, 수정은 동작(830)에 있는 시간 동안 차단되고, 다른 수정이 먼저 처리되지 않고는 실행될 수 없다. 각각의 수정이 처리된 후, 시스템은 수정된 콘텐츠 아이템의 리스트로부터 수정과 연관된 파일 식별자를 소거할 수 있다.

[0155] 도 9는 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한다. 도 9에 도시된 트리 데이터 구조는 클라이언트 디바이스에 저장될 수 있고, 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 관리될 수 있다. 예시를 위한 목적으로, 원격 트리(910)와 동기 트리(950)만이 도 9에 도시되고 설명된다. 로컬 트리에도 유사한 동작과 설명이 적용될 수 있다.

[0156] 원격 트리(910)는 1의 파일 식별자를 갖는 루트 노드(912), 5의 파일 식별자 및 "Foo"의 파일 이름을 갖는 노드(914), 6의 파일 식별자 및 "Bar"의 파일 이름을 갖는 노드(916), 및 7의 파일 식별자 및 "Bye"의 파일 이름을 갖는 노드(918)를 포함한다. 동기 트리는 1의 파일 식별자를 갖는 루트 노드(952)를 포함한다.

[0157] 도 9에 도시된 트리 데이터 구조에 기초하여, 시스템은 동작(810)에서 도 9에서 참조 부호(980)로 도시된 바와 같이 5, 6 및 7의 파일 식별자를 갖는 노드가 수정되었고 노드를 수정된 콘텐츠 아이템의 리스트에 추가된 것을 식별할 수 있다. 동작(815)에서, 시스템은 수정된 콘텐츠 아이템의 리스트에서 노드의 수정 리스트를 결정한다. 원격 트리(910)와 동기 트리(950)를 비교하여 알 수 있는 바와 같이, 노드(914, 916 및 918)가 원격 트리(910)에 추가되었다. 보다 구체적으로, 도 9에서 참조 부호(982)로 도시된 바와 같이, 6의 파일 식별자 및 "Bar" 이름을 갖는 노드(916)가 5의 파일 식별자를 갖는 노드(914)에 자식으로서 추가되었다. 이는 참조 부호(982)에서 "Add(6, 5, Bar)" 엔트리로 표현된다. 7의 파일 식별자와 이름 "Bye"를 갖는 노드(918)는 5의 파일 식별자를 갖는 노드(914)에 자식으로 추가되었다. 이는 참조 부호(982)에서 "Add(7, 5, Bye)" 엔트리로 표시된다. 5의 파일 식별자와 이름 "Foo"을 갖는 노드(914)는 1의 파일 식별자를 갖는 루트 노드(912)에 자식으로 추가되었다. 이는 참조 부호(982)에서 "Add(5,/root, Foo)" 엔트리로 표시된다.

[0158] 동작(820)에서, 시스템은 노드(914)의 추가 수정이 의존성을 갖지 않아서 차단되지 않은 것이라고 결정한다. 따라서, 시스템은 동작(825)에서 노드(914)와 연관된 수정(예를 들어, 참조 부호(982)에서 "Add(5,/root, Foo)" 엔트리로 표현된 수정)을 차단되지 않은 액션 리스트에 추가한다. 이것은 도 9의 참조 부호(984)에서 보인다. 한편, 참조 부호(982)에서 "Add(6, 5, Bar)" 및 "Add(7, 5, Bye)" 엔트리로 표현된 노드(916 및 918)의 수정은 "Add(5,/root, Foo)"로 표현된 수정이 먼저 일어나는 것에 의존한다. 다시 말해, 노드(916) 및/또는 노드(918)는 노드(914)가 추가될 때까지 추가될 수 없다. 따라서, 이러한 수정은 도 9의 참조 부호(986)로 도시된 차단된 액션 리스트에 포함된다.

[0159] 도 8의 방법(800)을 다시 참조하면, 동작(835)에서, 시스템은 차단되지 않은 액션 리스트로부터 수정 세트를 선택하고 선택된 수정 세트에 기초하여 동작 세트를 생성할 수 있다. 동작 세트는 서버 상태와 파일 시스템 상태를 수렴하도록 구성된다. 생성된 동작 세트는 차단되지 않은 리스트로부터 선택된 수정 세트에 의존한다. 예를 들어, 선택된 수정 세트가 도 9의 노드(914)와 연관된 추가 수정(예를 들어, 참조 부호(984)에서 "Add(5,/root, Foo)" 엔트리로 표시된 수정)을 포함하는 경우, 생성된 동작 세트는 콘텐츠 관리 시스템으로부터 추가된 콘텐츠 아이템을 검색하고 이를 클라이언트 디바이스의 로컬 파일 시스템에 추가하는 것을 포함할 수 있다.

[0160] 일부 실시형태에 따르면, 시스템은 차단되지 않은 액션 리스트로부터 모든 수정을 선택하여 하나 이상의 동작 세트를 생성할 수 있다. 그러나, 일부 시나리오에서, 차단되지 않은 리스트에서 수정의 수는 상당히 높을 수 있고 모든 수정을 처리하는데 필요한 컴퓨팅 자원(예를 들어, 메모리 및 처리 시간)이 상당하다. 이러한 기술적 부담을 줄이기 위해, 시스템은 증분적으로 처리하기 위해 차단되지 않은 액션 리스트에서 더 적은 수정 세트를 선택할 수 있다. 예를 들어, 시스템은 동작을 생성하기 위해 수정 중 제1 또는 상위 X개의 수 또는 백분율의 수정을 선택할 수 있다. 프로세스의 추가 반복에서, 차단되지 않은 리스트의 나머지 수정이 처리될 수 있다.

[0161] 일부 실시형태에서, 차단되지 않은 리스트의 수정은 처리를 위해 순위가 매겨질 수 있다. 수정은, 예를 들어 수정 유형(예를 들어, 삭제 수정이 추가 수정보다 우선됨), 수정과 연관된 메타데이터에 기초하여 순위화될 수 있다(예를 들어, 더 작은 크기의 콘텐츠 아이템의 수정을 추가하는 것이 더 큰 크기의 콘텐츠 아이템의 추가 수정보다 더 우선하고, 더 큰 크기의 콘텐츠 아이템의 삭제 수정이 더 작은 크기의 콘텐츠 아이템의 삭제 수정보다 우선한다 등).

[0162] 이들 순위 규칙은 시스템에 의해 저장될 수 있고, 콘텐츠 동기화를 위한 다양한 성능 목표를 달성하도록 설계될 수 있다. 예를 들어, 새로운 콘텐츠 아이템이 추가될 수 있기 전에 사용자를 위해 잠재적으로 제한된 저장 공간

을 최대한 확보하기 위해 삭제 수정이 추가 수정보다 우선 순위를 가질 수 있다. 가능한 한 빨리 추가된 콘텐츠 아이템의 수와 관련하여 많은 진전을 제공하기 위해 더 적은 콘텐츠 아이템의 추가가 더 큰 콘텐츠 아이템보다 우선 순위를 가질 수 있다.

[0163] 동작(835)에서, 시스템은 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스에 동작 세트를 제공할 수 있다. 위에서 언급한 바와 같이, 콘텐츠 관리 시스템에 의해 수행되는 액션과 연관된 수정은 클라이언트 디바이스에 반영되지 않았을 수 있다. 따라서, 이 시나리오에서, 시스템은 서버 상태와 파일 시스템 상태를 수렴하기 위해 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대해 동작하도록 구성된 클라이언트 동작 세트를 생성할 수 있고 이 클라이언트 동작 세트는 동작(835)에서 실행을 위해 클라이언트 디바이스에 제공될 수 있다.

[0164] 한편, 클라이언트 디바이스에 의해 수행된 액션과 연관된 수정은 콘텐츠 관리 시스템에 반영되지 않았을 수 있다. 따라서, 이 시나리오에서, 시스템은 서버 상태와 파일 시스템 상태를 수렴하기 위해 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대해 동작하도록 구성된 서버 동작 세트를 생성할 수 있고, 이 서버 동작 세트는 동작(835)에서 실행을 위해 관리 시스템 콘텐츠에 제공될 수 있다.

[0165] 일부 경우에, 두 경우는 모두 참일 수 있고, 클라이언트 동작 세트와 서버 동작 세트가 동작(835)에서 생성되고 의도된 수신자에 제공될 수 있다. 동작 세트는 또한 트리 제약이 유지되는 것을 보장하기 위해 다수의 검사를 포함할 수 있다. 예를 들어, 동작 세트는 도 6과 관련하여 논의된 바와 같이 다양한 충돌 또는 제약을 해결할 수 있다.

[0166] 동작 세트(들)가 의도된 수신자(들)에게 제공되면, 방법은 동작(805)으로 되돌아가서 새로운 수정 데이터를 기다릴 수 있다. 예를 들어, 도 9에 도시된 시나리오와 관련하여, 동작 세트는 콘텐츠 관리 시스템으로부터 노드(914)와 연관된 콘텐츠 아이템을 검색하고 이를 클라이언트 디바이스의 로컬 파일 시스템에 추가하는 것을 포함할 수 있다. 이것은 로컬 트리(도 9에 도시되지 않음)와 동기 트리(950)에서 노드(914)에 대응하는 노드를 추가하게 한다. 도 8의 프로세스(800)의 그 다음 반복에서, 참조 부호(982)에서 "Add(6, 5, Bar)" 및 "Add(7, 5, Bye)" 엔트리로 표현된 노드(916)와 노드(918)의 추가 수정은 그 부모 노드(914)가 동기 트리에 이미 추가되었기 때문에 더 이상 차단되지 않는다. 따라서, 참조 부호(982)에서 "Add(6, 5, Bar)" 및 "Add(7, 5, Bye)" 엔트리로 표현된 노드(916)와 노드(918)의 추가 수정은 차단되지 않은 액션 리스트에 추가될 수 있고, 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 하나 이상의 동작 세트를 생성하는 데 사용될 수 있다.

[0167] 동작 세트(들)는 서버 상태와 파일 시스템 상태를 증분적으로 수렴하기 위한 하나 이상의 단계를 제공할 수 있다. 증분적인 프로세스를 구현하는 것은 때때로 더 복잡할 수 있지만, 증분적인 프로세스는 처리 시간 단축 및 필요한 메모리 감소를 달성할 수 있다. 이러한 기술적 개선 및 다른 초기 기술적 개선은 자연스럽게 추가 기술적 개선으로 이어진다. 예를 들어, 처리 시간이 단축되기 때문에, 클라이언트 디바이스 또는 콘텐츠 관리 시스템으로부터 추가 변경이 특정 수정을 쓸모없거나 오래된 것으로 만들 가능성이 줄어든다.

[0168] 도 9와 관련하여, 콘텐츠 아이템, 수정, 액션 또는 파일 식별자의 다양한 그룹은 예시의 목적으로 리스트로서 기술된다. 다른 유형의 데이터 구조도 호환 가능하다. 예를 들어, 차단되지 않은 액션 리스트는 데이터를 정렬된 상태로 유지하고 로그 시간(logarithmic time)으로 탐색, 순차적 액세스, 삽입 및 삭제를 허용하기 위해 B-트리 데이터 구조로 구현될 수 있다.

[0169] 스케줄러

[0170] 일부 실시형태에서, 클라이언트 동기화 서비스는 서버 상태와 파일 시스템 상태를 수렴하고 실행을 위해 콘텐츠 관리 시스템 또는 클라이언트 디바이스에 동작을 제공하도록 구성된 동작 세트 또는 동작 시퀀스를 생성할 수 있다. 그러나 일부 시나리오에서 클라이언트 디바이스의 파일 시스템 또는 콘텐츠 관리 시스템의 변경은 동작 세트가 실행 중인 동안 생성된 동작 세트가 오래된 것이거나 더 이상 쓸모없게 할 수 있다. 다양한 실시형태는 이들 및 다른 기술적 문제에 대한 기술적 해결책을 제공하는 것에 관한 것이다. 예를 들어, 클라이언트 동기화 서비스는 클라이언트 디바이스의 파일 시스템 또는 콘텐츠 관리 시스템의 변경 사항을 모니터링하고 클라이언트 디바이스 및/또는 콘텐츠 관리를 필요에 따라 업데이트하도록 구성될 수 있다. 또한, 클라이언트 동기화 서비스는 동작의 동시 실행을 허용함으로써 성능을 개선하고 처리 시간을 감소시키도록 구성될 수 있다.

[0171] 일부 실시형태에 따르면, 도 2에 도시된 클라이언트 동기화 서비스(156)의 플래너(225)는 비-정렬된 동작 세트로 구성되는 동작 계획 또는 동작 계획들을 생성할 수 있다. 계획 내의 모든 동작은 의존성이 없어서, 별개의 스레드 또는 임의의 순서로 동시에 실행될 수 있다. 일부 실시형태에 따른, 계획에서의 동작은 상태와 트리 데이터 구조를 수렴하기 위해 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스에 의해 취해질 수 있는 추상 명령

이다. 예시적인 명령은 콘텐츠 아이템의 원격 또는 로컬 추가, 콘텐츠 아이템의 원격 또는 로컬 삭제, 콘텐츠 아이템의 원격 또는 로컬 편집, 또는 콘텐츠 아이템의 원격 또는 로컬 이동을 포함할 수 있다.

[0172] 도 2에 도시된 클라이언트 동기화 서비스(156)의 스케줄러(230)는 플래너(225)로부터 동작 계획을 수신하고, 계획의 동작 실행을 관리하고, 계획이 업데이트 또는 변경되었는지 여부를 결정하고, 업데이트되거나 변경된 계획의 실행을 관리하도록 구성될 수 있다. 예를 들어, 스케줄러(230)는 파일 시스템 인터페이스(205) 및 서버 인터페이스(210)와 협력하여 계획에서의 동작을 구현하는 데 필요한 작업 및 단계를 실행할 수 있다. 이것은 파일 시스템 또는 콘텐츠 관리 시스템으로부터 확인을 수신하거나, 또는 네트워크 연결이 없을 때 또는 콘텐츠 아이템이 일부 다른 애플리케이션에 의해 잠겨 있을 때 재시도를 처리하는 등의 오류 처리 활동을 수신하는 것을 포함할 수 있다.

[0173] 각각의 동작은 작업이라고 지칭되는 스크립트 또는 스레드에 의해 구현될 수 있다. 작업은 연관된 동작의 적용을 조정하고, 동작을 구현하는데 필요한 하나 이상의 단계를 포함할 수 있다. 예를 들어, "로컬 추가 동작"은, 콘텐츠 아이템이 클라이언트 디바이스의 로컬 파일 시스템에 추가되어서, 콘텐츠 아이템이 서버 상태와 파일 시스템 상태를 동기화하기 위해 콘텐츠 관리 시스템에 추가되어야 한다는 것을 나타낼 수 있다. 따라서, 로컬 추가 동작은 로컬 추가 동작을 구현하는데 필요한 하나 이상의 단계를 포함하는 "로컬 추가 작업"과 연관될 수 있다. 단계는 콘텐츠 관리 시스템에 새로운 콘텐츠 아이템을 통지하는 것, 콘텐츠 아이템을 하나 이상의 데이터 블록으로 콘텐츠 관리 시스템에 업로드하는 것, 콘텐츠 관리 시스템에 의해 모든 데이터 블록이 수신되었음을 확인하는 것, 콘텐츠 아이템이 손상되지 않은 것을 보장하는 것, 콘텐츠 아이템에 대한 메타데이터를 콘텐츠 관리 시스템에 업로드하는 것, 및 콘텐츠 아이템을 콘텐츠 관리 시스템의 적절한 위치에 추가하는 것을 커밋하는 것 중 하나 이상을 포함할 수 있다.

[0174] 작업은 실행을 시작하고, 다른 이벤트의 완료를 기다리는 동안 잘 규정된 지점에서 일시 중단하고, 이벤트가 발생하면 재개하고, 최종 종료될 수 있다. 일부 실시형태에 따르면, 스케줄러(230)는 작업을 취소, 재생성 또는 교체하도록 구성된다. 예를 들어, 서버 상태 또는 파일 시스템 상태에 대한 변경에 기초하여, 작업은 실행되기 전에 실효(stale)될 수 있고, 스케줄러(230)는 실행되기 전에 실효된 작업을 취소할 수 있다.

[0175] 전술한 바와 같이, 플래너(225)는 트리 데이터 구조의 세트(예를 들어, 원격 트리, 동기 트리 및 로컬 트리)에 기초하여 동작 계획을 생성할 수 있다. 시간이 지남에 따라, 플래너(225)는 트리 데이터 구조의 상태에 기초하여 동작 계획을 계속 생성한다. 트리 데이터 구조가 서버 상태와 파일 시스템 상태의 상태를 반영하도록 변경되면, 플래너(225)는 또한 이전 계획과 다른 새로운 업데이트된 계획을 생성할 수 있다. 스케줄러(230)는 플래너(225)에 의해 생성된 각각의 동작 계획을 실행한다.

[0176] 일부 시나리오에서, 후속 계획의 동작의 변경은 실행 중인 이전의 계획의 동작과 의도치 않은 거동 충돌을 야기할 수 있다. 예를 들어, 제1 계획의 동작이 실행되고 있기 때문에 제2 계획에서 하나 이상의 동작이 취소된다(존재하지 않는다). 설명을 위해, 도 10은 시간(t1)에서, 원격 트리로 표현된 서버 상태와 로컬 트리로 표현된 파일 시스템 상태가 원격 트리, 동기 트리 및 로컬 트리가 모두 일치하는 것으로 도시된 바와 같이 동기화되는 예시적인 시나리오를 도시한다. 이 동기화된 상태에 기초하여, 플래너(225)는 t1에서 동작을 갖지 않는 계획(예를 들어, 빈 계획)을 생성할 수 있다.

[0177] 클라이언트 디바이스의 사용자는 로컬 파일 시스템으로부터 콘텐츠 아이템(A)을 삭제하거나, 클라이언트 동기화 서비스(156)에 의해 관리되는 폴더 밖으로 콘텐츠 아이템(A)을 이동할 수 있으며, 이는 시간(t2)에 로컬 트리로부터 노드(A)를 제거하는 것에 의해 반영된다. 플래너(225)는 시간(t2)에서 트리 데이터 구조의 상태에 기초하여 동작(LocalDelete(A))을 포함하는 계획을 생성할 수 있다. 스케줄러(230)는 LocalDelete(A) 동작을 구현하는데 필요한 작업 또는 단계를 개시할 수 있다. 이 단계는 콘텐츠 아이템(A)을 삭제하라는 명령을 콘텐츠 관리 시스템에 전송하는 단계를 포함할 수 있다.

[0178] 콘텐츠 아이템(A)을 삭제하라는 명령이 콘텐츠 관리 시스템으로 전송된 후, 클라이언트 디바이스의 사용자는 콘텐츠 아이템(A)의 삭제를 실행 취소하거나 콘텐츠 아이템(A)을 이전의 위치로 다시 이동할 수 있다. 로컬 트리는 시간(t3)에서 이 새로운 액션에 기초하여 업데이트되고, 플래너는 동작을 갖지 않는 비어 있는 새로운 계획을 생성할 수 있다. 다시 한번, 트리 데이터 구조가 일치하고 시스템은 시간(t3)에서 동기화된 상태에 있다.

[0179] 그러나, 콘텐츠 아이템(A)을 삭제하라는 명령은 콘텐츠 관리 시스템으로 전송되었기 때문에, 콘텐츠 관리 시스템은 서버 상태로부터 콘텐츠 아이템(A)을 삭제한다. 스케줄러(230)는 콘텐츠 아이템(A)의 삭제를 취소하려고 시도할 수 있지만, 명령은 콘텐츠 관리 시스템에 의해 이미 전송되고 완료되었을 수 있다. 서버에서의 이러한

변경은 클라이언트 동기화 서버(156)로 전달되고, 클라이언트 동기화 서버는 시간(t4)에서 노드(A)를 삭제함으로써 원격 트리를 업데이트한다. 플래너(225)는 원격 트리에서의 변경 사항, 및 원격 트리와 동기 트리 사이의 차이를 인지하고, 콘텐츠 아이템(A)이 서버 상태에서 제거되었다고 결정할 수 있다. 따라서, 플래너(225)는 시간(t4)에서 RemoteDelete(A) 동작을 갖는 계획을 생성한다. 서버 상태와 파일 시스템 상태를 동기화하기 위해 콘텐츠 아이템(A)은 결국 클라이언트 디바이스와 로컬 트리로부터 삭제된다.

[0180] 문제가 되는 것으로는, 서버 상태로부터 콘텐츠 아이템(A)의 제거, RemoteDelete(A) 동작의 생성, 및 파일 시스템 상태로부터 콘텐츠 아이템(A)의 최종 제거는 모두 의도된 것이 아니며, 사용자를 위한 라인 아래로 추가적인 문제를 야기할 수 있다. 또한 일부 경우에 애플리케이션이나 프로세스가 또한 콘텐츠 아이템에 액세스할 수 있으며, 의도치 않은 동기화 거동은 추가적인 기술적 문제를 야기할 수 있다. 다양한 실시형태는 서버 상태와 파일 시스템 상태 사이에서 콘텐츠 아이템의 동기화에 의도치 않은 결과가 발생하는 것을 방지하는 것에 관한 것이다.

[0181] 일부 실시형태에 따르면, 더 이상 동작 계획에 없는 실효된 동작에 대한 작업을 취소할 때, 스케줄러(230)는 다른 작업의 실행을 개시하려고 진행하기 전에 취소가 완료되기를 기다릴 수 있다. 예를 들어, 스케줄러(230)는 다른 작업을 진행하기 전에 클라이언트 디바이스 또는 콘텐츠 관리 시스템으로부터 취소 확인을 수신하기를 기다릴 수 있다. 스케줄러(230)는 작업이 개시되었는지 여부를 결정할 수 있고, 작업이 개시되지 않은 경우, 스케줄러는 작업을 취소하고, 작업이 더 이상 실행을 기다리고 있지 않음을 확인할 수 있다. 작업이 개시된 경우, 확인은 클라이언트 디바이스 또는 콘텐츠 관리 시스템으로부터 오고, 취소된 작업과 연관된 모든 단계가 실행 취소되었음을 스케줄러에게 통지할 수 있다. 일부 구현예에 따르면, 스케줄러(230)는 작업이 일단 개시되었다면 작업의 취소를 허용하지 않는다. 이것은 모든 작업 또는 작업 또는 작업 유형(예를 들어, 서버 상태와 동기화하기 위해 파일 시스템 상태의 업데이트를 콘텐츠 관리 시스템으로 전송하는 커밋 작업)의 특정 서브 세트에 대해 그러할 수 있다.

[0182] 성능을 개선하고 작업의 동시 실행과 작업의 취소를 허용하기 위해, 스케줄러(230)는 제1 동작 계획과 업데이트된 제2 동작 계획 사이의 차이에 기초하여 작업의 실행 및 작업의 취소를 관리하도록 구성될 수도 있다. 도 11은 본 기술의 다양한 실시형태에 따른 2개의 동작 계획의 예시적인 벤 다이어그램(1100)을 도시한다. 플래너(225)는 제1 동작 세트를 갖는 계획 1(1110)을 생성하고, 트리 데이터 구조에 대한 업데이트를 수신하고, 제2 동작 세트를 갖는 업데이트된 계획 2(1120)를 생성할 수 있다.

[0183] 계획 1(1110) 및 계획 2(1120)는 벤 다이어그램(1100)의 부분(1130)으로 표현된 다수의 공통 동작을 공유할 수 있다. 계획 1(1110) 및 계획 2(1120)는 또한 공통적이지 않은 다수의 동작을 공유할 수 있다. 예를 들어, 계획 2(1120)에 있지 않은 계획 1(1110)의 동작은 플래너(225)에 의해 검출된 트리 구조에 대한 업데이트에 기초하여 실효되고 더 이상 현재 상태가 아니다. 계획 1(1110)의 이 실효된 동작은 벤 다이어그램(1100)의 부분(1140)으로 표현된다. 계획 1(1110)에 없는 계획 2(1120)의 새로운 동작은 부분(1150)으로 표현된다. 계획 1(1110)과 계획 2(1120) 사이의 차이 및 공통 부분을 나타내는 부분(1130, 1140 및 1150) 각각은 트리 데이터 구조에 반영된 서버 상태와 파일 시스템 상태에 대한 업데이트에 따라 동작을 포함하지 않거나 많은 동작을 포함할 수 있다.

[0184] 부분(1140)의 동작은 더 이상 가장 최근의 계획이 아니기 때문에, 스케줄러(230)는 이 동작과 연관된 작업을 취소할 수 있다. 의도치 않은 동기화 거동이 일어나는 것을 방지하기 위해, (예를 들어, 부분(1150)에서) 계획 1에 없는 계획 2의 동작과 연관된 작업은 부분(1140)의 동작과 연관된 작업의 취소가 완료될 때까지 연기된다. 그러나, 각각의 계획의 동작들이 동시에 실행될 수 있도록 구성되기 때문에, 부분(1130)으로 표현된 계획 1과 계획 2의 교점의 동작과 연관된 작업은 완료를 기다리지 않고 부분(1140)의 동작과 연관된 작업의 취소와 동시에 실행될 수 있다. 부분(1140)과 연관된 작업의 동시 취소 및 부분(1130)과 연관된 작업의 실행을 허용함으로써, 이용 가능한 컴퓨팅 자원의 보다 효율적인 사용 및 처리 시간의 단축이 달성될 수 있다.

[0185] 도 12는 본 기술의 다양한 실시형태에 따른, 동작 계획의 변경을 관리하기 위한 예시적인 방법을 도시한다. 본 명세서에 기술된 방법 및 프로세스가 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(1200)은 예를 들어 클라이언트 디바이스에서 실행되는 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0186] 시스템은 콘텐츠 관리 서비스와 연관된 콘텐츠 아이템과 관련하여 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스로부터 업데이트를 수신하도록 구성될 수 있다. 예를 들어, 시스템은 콘텐츠 관리 서비스에 의해 저장된 콘텐츠 아이템에 대한 서버 수정 데이터를 수신하고, 서버 수정 데이터에 기초하여 원격 트리를 업데이트할 수 있

다. 원격 트리는 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템의 서버 상태를 나타낸다. 시스템은 또한 클라이언트 디바이스에 저장된 콘텐츠 아이템에 대한 클라이언트 수정 데이터를 수신하고, 클라이언트 수정 데이터에 기초하여 로컬 트리를 업데이트할 수 있다. 로컬 트리는 클라이언트 디바이스에 저장된 콘텐츠 아이템의 파일 시스템 상태를 나타낸다.

[0187] 동작(1205)에서, 시스템은 콘텐츠 관리 시스템과 연관된 서버 상태와 클라이언트 디바이스와 연관된 파일 시스템 상태를 수렴하도록 구성된 제1 동작 세트를 수신할 수 있다. 예를 들어, 시스템은 동기 트리와 원격 트리 사이 또는 동기 트리와 로컬 트리 사이의 차이를 식별하고, 트리들 사이의 임의의 차이에 기초하여 제1 동작 세트를 생성할 수 있다. 동기 트리는 서버 상태와 파일 시스템 상태 간의 알려진 동기화 상태를 나타낸다.

[0188] 시스템은 제1 동작 세트를 구현하기 시작할 수 있다. 예를 들어, 일부 경우에, 동작은 실행을 위해 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스로 전송될 준비가 된 포맷으로 되어 있다. 다른 경우에 동작은 시스템에 의해 관리될 수 있는 하나 이상의 작업, 스크립트 또는 실행 스레드로 변환될 수 있다. 시스템은 서버 상태와 파일 시스템 상태를 수렴하기 위해 작업, 스크립트 또는 실행 스레드에 따라 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스와 인터페이스할 수 있다.

[0189] 이 시간 동안, 시스템은 콘텐츠 관리 서비스와 연관된 콘텐츠 아이템과 관련하여 콘텐츠 관리 시스템 및/또는 클라이언트 디바이스로부터 수정 데이터를 계속 수신할 수 있다. 수정 데이터에 기초하여, 시스템은 원격 트리 또는 로컬 트리를 업데이트하고, 트리 데이터 구조에 대한 업데이트에 기초하여 제2 동작 세트를 생성할 수 있다. 동작(1210)에서, 시스템은 제2 동작 세트를 수신할 수 있다.

[0190] 동작(1215)에서, 시스템은, 만약 있다면, 제2 동작 세트에 있지 않은 제1 동작 세트의 제1 동작을 식별한다. 시스템이 제2 동작 세트에 없는 제1 동작 세트의 동작을 찾으면, 이 동작은 수정 데이터에 제시된 변경으로 인해 실효되고 오래된 것일 수 있다. 따라서, 시스템은 동작(1220)에서 제1 동작의 취소를 개시한다. 제1 동작의 취소는 다수의 단계, 단계에 대한 확인 수신의 수, 및 상당한 양의 처리 시간을 포함할 수 있다.

[0191] 동작(1225)에서, 시스템은 만약 있는 경우 제1 동작 세트와 제2 동작 세트에 모두 포함되는 제2 동작을 식별한다. 시스템이 제1 동작 세트와 제2 동작 세트 모두에서 동작을 발견하면 이 동작은 수정 데이터에 지정된 변경 사항에도 불구하고 여전히 유효할 수 있다. 또한, 두 동작 세트 내 동작이 세트 내 다른 동작에 대해 동시에 또는 임의의 순서로 실행될 수 있도록 구성되기 때문에, 제2 동작은 제1 동작이 취소되는 동안 계속 실행될 수 있다. 따라서, 시스템은 제1 동작의 취소 완료를 기다리지 않고 동작(1230)에서 제2 동작의 실행을 개시한다.

[0192] 동작(1235)에서, 시스템은 만약 있는 경우 제2 동작 세트에는 있지만 제1 동작 세트에는 없는 제3 동작을 식별한다. 시스템이 제1 동작 세트에 없는 제2 동작 세트의 동작을 발견하면, 이 동작은 수정 데이터에 지정된 변경으로 인한 새로운 동작일 수 있다. 의도치 않은 결과가 발생하는 것을 방지하기 위해 시스템은 제1 동작이 취소 완료되는 동안 대기를 개시한다. 동작(1240)에서, 시스템은 제1 동작이 취소 완료되었다고 결정하고, 그 결과 동작(1245)에서 제3 동작의 실행을 개시할 수 있다.

[0193] 로컬 트리 업데이트

[0194] 전술한 바와 같이, 로컬 트리는 클라이언트 디바이스의 로컬 파일 시스템에 저장된 콘텐츠 아이템에 대한 파일 시스템 상태를 반영하도록 구성된다. 예를 들어, 도 2의 클라이언트 동기화 서비스(156)의 파일 시스템 인터페이스(205)는 클라이언트 디바이스의 로컬 파일 시스템을 변경하고(예를 들어, 하나 이상의 콘텐츠 아이템을 추가, 삭제, 이동, 편집 또는 이름 변경), 로컬 파일 시스템의 변경을 감지하고, 로컬 파일 시스템에 대한 변경 사항에 기초하여 로컬 트리를 업데이트하도록 구성된다. 변경은 파일 시스템의 사용자 액션에 의해, 클라이언트 디바이스에서 실행 중인 제3자 애플리케이션 또는 파일 시스템 상태를 서버 상태와 동기화하는 클라이언트 동기화 서비스에 의해 발생될 수 있다.

[0195] 본 기술의 다양한 실시형태는 로컬 파일 시스템의 변경에 기초하여 로컬 트리를 업데이트하기 위한 다양한 기술적 해결책을 제공한다. 로컬 트리는 다른 트리 데이터 구조와 함께 다양한 실시형태에서 클라이언트 디바이스와 콘텐츠 관리 시스템 간의 동기화 프로세스에 중요하다. 예를 들어, 로컬 트리에 대한 업데이트가 이루어지면 나머지 시스템은 업데이트에 반응하고, 일부 경우 로컬 트리에 대한 변경 사항이 동기화되어 콘텐츠 관리 시스템의 서버 상태에 적용될 수 있다. 따라서 로컬 트리가 어떻게 업데이트되었는지를 주목하는 것이 중요하다.

[0196] 예를 들어, 사용자가 파일을 A.txt로부터 B.txt로 이름을 변경하는 경우, 일부 경우에, 시스템은 콘텐츠 아이템 A.txt의 삭제 및 콘텐츠 아이템 B.txt의 추가를 검출할 수 있다. 이것은 로컬 트리에서 A.txt에 대한 노드를 삭제하고 B.txt에 대한 노드를 추가시킬 수 있다. 그러나 이것으로 이름이 변경된 콘텐츠 아이템의 노드가 로컬

트리에 일부 시간 동안 존재하지 않는 경우가 발생한다. 이것은 클라이언트 디바이스, 클라이언트 애플리케이션 및/또는 클라이언트 동기화 서비스가 B.txt에 대한 노드가 추가되어 사용자의 콘텐츠 아이템이 손실되기 전에 종료, 실패 또는 재부팅될 수 있기 때문에 데이터 무결성을 크게 손상시킬 수 있다. 사용자의 콘텐츠 아이템의 손실은 콘텐츠 관리 시스템에서 서버 상태와 동기화될 수 있다. 사용자가 콘텐츠 아이템을 하나의 위치로부터 다른 위치로 이동시키는 것과 연관된 유사한 위험이 있다.

[0197] 추가적으로, 로컬 파일 시스템에 대한 변경 사항은 순서가 맞지 않게 검출될 수 있으며, 사용자 또는 애플리케이션에 의한 단일 액션과 모두 반드시 관련된 것은 아닌 많은 수의 변경을 포함할 수 있다. 로컬 파일 시스템에 많은 변경이 이루어지는 동안 클라이언트 애플리케이션은 꺼져 있거나 실행되고 있지 않을 수도 있다. 시작 시 클라이언트 애플리케이션은 로컬 파일 시스템을 크롤링하고, 이를 로컬 트리와 비교하고, 클라이언트 애플리케이션이 꺼져 있는 동안 발생한 로컬 파일 시스템의 변경 사항이 무엇인지를 결정할 수 있다. 이러한 변경 사항은 시간적으로 순서가 맞지 않을 수 있다. 이러한 요인은 로컬 트리를 신중하게 업데이트하지 않으면 의도치 않은 동기화 거동을 초래할 수도 있다.

[0198] 트리 데이터 구조의 무결성을 보장하고 의도치 않은 동기화 거동이 일어나는 것을 방지하기 위해 제약 세트가 사용될 수 있다. 이 제약은 예를 들어 (1) 트리의 모든 노드가 파일 식별자(fileID)와 연관되어 있다는 것, (2) 트리의 모든 노드가 고유한 파일 식별자를 가지고 있다는 것, (3) 비어 있지 않은 부모 노드는 삭제될 수 없다는 것, (4) 삭제된 노드는 클라이언트 동기화 서비스에 의해 관리되는 위치로부터 실제로 삭제(그리고 단순히 이동되는 것이 아님)되거나 제거된다는 것, (5) 모든 형제 노드는 경우에 관계없이 고유한 파일 이름을 갖는다는 것, (6) 모든 노드는 존재하는 부모를 가져야 한다는 것, 및/또는 (7) 모든 형제 노드는 부모 파일 ID(DirFileID)에 동의한다는 것을 포함할 수 있다. 일부 구현예에서, 상기 제약의 부분 세트가 사용될 수 있거나, 대안적인 또는 추가적인 제약이 사용될 수 있거나, 또는 조합이 사용될 수 있다. 제약의 서브 세트는 모든 트리 데이터 구조에 적용되거나 또는 트리 데이터 구조의 서브 세트에만 적용될 수 있는 반면, 상이한 제약 세트 또는 제약 세트들은 다른 트리 데이터 구조에 적용될 수 있다.

[0199] 로컬 파일 시스템에 대한 변경이 검출될 때, 변경은 제약 세트에 대해 검사될 수 있다. 변경 사항이 제약 세트와 일치하면 로컬 파일 시스템의 변경 사항에 기초하여 로컬 트리가 업데이트될 수 있다. 변경 사항이 제약 중 하나를 위반하는 경우 제약은 추가 조건이 충족될 것을 요구할 수 있다. 예를 들어, 제약은 변경 사항이 로컬 트리에 적용되거나, 하나 이상의 교정 단계가 수행되거나, 또는 이들의 조합이 수행되기 전에 추가 경로가 관찰되거나 파일 이벤트가 발생할 것을 요구할 수 있다. 특정 제약(예를 들어, 교정 단계 수행, 추가 경로 관찰 또는 파일 이벤트 발생)을 충족시키기 위한 액션이 일어날 때 다른 제약이 위반될 수 있다. 따라서, 제약 세트는 모든 제약이 충족될 때까지 지속적으로 검사될 수 있다. 제약이 충족되면 파일 이벤트와 연관된 변경 사항이 로컬 트리에 적용될 수 있다.

[0200] 파일 이벤트는 변경이 로컬 파일 시스템에서 검출된 것에 응답하여 클라이언트 동기화 서비스(156)에 의해 검출될 수 있다. 각각의 파일 이벤트는 콘텐츠 아이템(예를 들어, 콘텐츠 아이템에 대한 파일 식별자) 및 이벤트 유형(예를 들어, 추가, 이동, 삭제 또는 편집 이벤트 유형)과 연관될 수 있다. 각 파일 이벤트는 연관된 콘텐츠 아이템의 경로 또는 위치를 지정하는 경로와 연관될 수도 있다. 검출된 파일 이벤트와 연관된 경로는 클라이언트 동기화 서비스에 의해 관찰되는 경로 세트를 채울 수 있다. 그러나 경우에 따라 하나 이상의 제약 위반으로 인해 파일 이벤트에 대응하지 않는 경로가 관찰될 수 있다.

[0201] 도 13은 본 기술의 다양한 실시형태에 따른 예시적인 시나리오를 도시한다. 특히, 도 13은 파일 이벤트(1315)가 검출될 때 로컬 트리(1310)의 현재 상태를 도시한다. 예를 들어, 클라이언트 동기화 서비스는 파일 시스템을 로컬 트리(1310)와 비교하고, 콘텐츠 아이템이 /root/a/b/c.txt 경로의 파일 시스템에 존재하지만 /root/a/b/c.txt에서 콘텐츠 아이템에 대한 노드가 로컬 트리(1310)에 존재하지 않음을 발견할 수 있다. 따라서, /root/a/b/c.txt에서 노드의 추가가 로컬 트리(1310)에 필요하다는 것을 나타내는 파일 이벤트(1315)가 생성될 수 있다.

[0202] 클라이언트 동기화 서비스는 파일 이벤트(1315)를 업데이트를 위해 관찰된 경로 세트에 추가하고, 관찰된 경로(1320)가 제약 세트와 일치하는지 여부를 결정하고, 관찰된 경로(1320) 또는 파일 이벤트(1315)가 세트 내 제약 중 하나를 위반하는 것을 발견할 수 있다. 도 13에 도시된 시나리오에서, 관찰된 경로(1320)는 "모든 노드가 존재하는 부모를 가져야 한다는" 제약을 위반한다. 보다 구체적으로, /root/a/b/c.txt에 추가될 노드의 부모도 존재하지 않고 노드의 조부모도 존재하지 않는다. 따라서 변경 사항이 로컬 트리에 적용되기 전에 추가 경로(부모 및 조부모 노드)가 관찰되어야 한다.

[0203] 클라이언트 동기화 서비스는 추가 파일 이벤트를 검출하고, 이를 업데이트를 위해 관찰된 경로 세트에 추가할 수 있다. 예를 들어, 클라이언트 동기화 서비스는 /root/a 파일 이벤트 및 add/root/a/b 파일 이벤트를 검출하고, /root/a 경로 및 add/root/a/b 경로를 관찰된 경로 세트에 추가할 수 있다. 이러한 경로가 관찰되면 위반된 제약이 충족된다(그리고 다른 제약은 위반되지 않는다). 그 결과, 관찰된 업데이트 파일 이벤트 모두가 로컬 트리에 적용될 수 있다. 보다 구체적으로, 노드는 /root/a에 추가될 수 있고, 노드는 /root/a/b에 추가될 수 있으며, 노드는 /root/a/b/c.txt에 추가될 수 있다. 따라서 클라이언트 동기화 서비스는 원자 단위 업데이트(atomic update) 또는 단일형 업데이트를 위해 함께 관련된 파일 이벤트를 그룹화한다. 보다 상세하게 설명되는 바와 같이, 로컬 트리에 대한 원자 단위 업데이트를 위해 함께 관련된 파일 이벤트를 그룹화하면 트리 데이터 구조의 무결성이 증가하고, 의도치 않은 동기화 거동이 발생하는 것이 방지되며, 로컬 트리에서 중간 상태가 발생하는 것이 방지된다.

[0204] 도 14는 본 기술의 다양한 실시형태에 따라 로컬 트리를 업데이트하기 위한 예시적인 방법을 도시한다. 본 명세서에 기술된 방법 및 프로세스가 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(1400)은 예를 들어 클라이언트 디바이스에서 실행되는 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0205] 동작(1405)에서, 시스템은 파일 이벤트를 검출하고, 파일 이벤트와 연관된 경로를 관찰된 경로 세트에 추가한다. 예를 들어, 시작 시 시스템은 클라이언트 디바이스의 파일 시스템을 크롤링하고, 파일 시스템에 대한 정보를 수집하고, 수집된 정보를 파일 시스템의 마지막 알려진 상태를 나타내는 로컬 트리와 비교할 수 있다. 시스템은 클라이언트 디바이스의 파일 시스템과 로컬 트리 사이의 차이를 식별하고, 식별된 차이에 기초하여 다수의 파일 이벤트를 생성할 수 있다. 대안적으로 또는 추가적으로, 시스템은 런타임 동안 파일 시스템을 모니터링하고, 로컬 트리에 반영되지 않은 파일 시스템에 이루어진 변경 사항을 검출하고, 파일 시스템에 대해 검출된 변경 사항에 기초하여 파일 이벤트를 생성할 수 있다. 생성된 파일 이벤트는 시스템에 의해 이루어진 파일 시스템에 대한 관찰로 고려될 수 있다.

[0206] 동작(1410)에서, 시스템은 관찰된 경로 세트를 로컬 트리 제약 세트와 대비 검사하여, 관찰된 경로 중 임의의 것이 로컬 트리 제약을 위반하는지 여부를 결정한다. 관찰된 경로 중 그 어느 것도 로컬 트리 제약 세트의 임의의 제약을 위반하지 않으면, 동작(1435)에서 관찰된 경로 세트는 로컬 트리를 업데이트하는 데 사용될 수 있다.

[0207] 제약의 각 위반은 제약을 충족시키도록 구성된 교정과 연관될 수 있다. 예를 들어, "모든 노드는 존재하는 부모를 가져야 한다"는 제약의 위반은 도 13에 도시된 바와 같이 부모 노드의 추가가 관찰되어야 한다는 요구 조건과 연관될 수 있다.

[0208] 다른 경우에, 제약의 위반은 위반을 해결하고 제약을 충족시키기 위해 액션을 취할 것을 요구할 수 있다. 예를 들어, 사용자가 파일 시스템의 콘텐츠 아이템을 복사하고 존재하는 콘텐츠 아이템의 새로운 사본을 생성할 때, 새로운 콘텐츠 아이템은 원래 콘텐츠 아이템과 동일한 파일 식별자를 가질 수 있다. 시스템은 새로운 사본의 추가를 관찰할 수 있지만, 이 새로운 사본은 "트리의 모든 노드가 고유한 파일 식별자를 갖는다"는 제약을 위반한다. 이 제약의 위반은, 콘텐츠 아이템에 대한 새로운 파일 식별자를 요청하고, 새로운 콘텐츠 식별자를 콘텐츠 아이템에 할당하여, 로컬 트리가 업데이트되기 전에 위반을 해결하고 제약을 충족시키는 교정 단계와 연관될 수 있다. 따라서 로컬 트리는 어떤 지점에서도 임의의 제약을 위반하는 상태가 있지 않다.

[0209] 다른 예에서, 사용자는 비록 대소 문자가 상이하지만 파일 이름이 이미 존재하는 파일 시스템의 위치에 새로운 콘텐츠 아이템을 생성할 수 있다. 예시를 위해, 파일 "a.txt"가 동일한 파일 시스템 경로로 이미 존재하는 경우 사용자는 "A.txt"라는 파일을 생성할 수 있다. 클라이언트 디바이스의 동작 시스템은 이를 허용할 수 있지만 클라이언트 동기화 서비스는 이를 허용하지 않을 수 있다. 시스템은 새로운 콘텐츠 아이템의 추가를 관찰할 수 있지만, 이 새로운 콘텐츠 아이템은 "모든 형제 노드는 대소 문자에 관계없이 고유한 파일 이름을 갖는다"는 제약을 위반한다. 이 제약을 위반하는 것은 대소 문자 충돌(대소 문자 충돌)이 있음을 나타내기 위해 새로운 콘텐츠 아이템의 이름을 편집하는 교정 단계와 연관될 수 있다. 예를 들어, "A.txt" 파일은 "A(대소 문자 충돌).txt"로 이름이 변경되어, 위반 사항을 해결하고 제약을 충족시킬 수 있다. 파일 이벤트 및 경로는 관찰된 경로 세트로부터 제거될 수 있으며, 프로세스는 "A(대소 문자 충돌).txt" 콘텐츠 아이템의 추가를 위한 새로운 파일 이벤트가 검출되거나 또는 "A.txt"에 대한 파일 이벤트가 새로운 이름 "A(대소 문자 충돌).txt"를 반영하도록 업데이트되도록 재시작된다.

[0210] 하나 이상의 관찰된 경로가 하나 이상의 제약을 위반하는 경우, 시스템은 위반된 제약이 동작(1415)에서 교정

액션이 취해질 것을 요구하는지 여부 또는 위반된 제약이 동작(1425)에서 추가 경로가 관찰될 것을 요구하는지 여부를 결정할 수 있다. 추가 교정 액션이 요구되는 경우, 동작(1420)에서, 시스템은 추가 교정 액션을 실행할 수 있다. 추가 경로가 관찰되는 경우, 동작(1430)에서, 시스템은 추가 파일 이벤트를 검출하고, 동작(1430)에서 파일 이벤트와 연관된 경로를 관찰된 경로 세트에 추가할 수 있다.

[0211] 그 후, 프로세스는 동작(1410)으로 되돌아가서 관찰된 경로 세트가 로컬 트리 제약을 위반하는지 여부를 결정한다. 경우에 따라 교정 액션 실행, 새로운 파일 이벤트 검출, 또는 관찰된 경로 세트에 경로의 추가는 로컬 트리의 업데이트를 수행하기 전에 해결되어야 하는 하나 이상의 제약에 대한 새로운 위반을 야기할 수 있다. 따라서, 로컬 트리 제약에 대한 위반이 더 이상 존재하지 않을 때까지 프로세스가 반복될 수 있다. 그 후 프로세스는 동작(1435)으로 진행할 수 있으며, 여기서 시스템은 관찰된 경로 세트에 기초하여 로컬 트리를 업데이트할 수 있다.

[0212] 이동 또는 이름 변경을 갖는 로컬 트리 업데이트

[0213] 일부 구현예에 따르면, 로컬 파일 시스템 상의 콘텐츠 아이템에 대한 이동 또는 이름 변경 동작은 추가적인 기술적 문제를 야기할 수 있다. 예를 들어, 파일 또는 폴더와 같은 콘텐츠 아이템이 사용자 또는 애플리케이션에 의해 오래된 위치로부터 새로운 위치로 이동하는 경우, 오래된 위치로부터 콘텐츠 아이템의 삭제 및 새로운 위치에 새로운 콘텐츠 아이템의 추가로서 파일 시스템 또는 클라이언트 애플리케이션에 동작이 나타날 수 있다. 유사하게, 오래된 파일 이름으로부터 새로운 파일 이름으로 콘텐츠 아이템의 이름 변경은 오래된 파일 이름을 갖는 콘텐츠 아이템을 삭제하고 새로운 파일 이름을 갖는 새로운 콘텐츠 아이템을 추가하는 것으로 나타날 수 있다. 또한 콘텐츠 아이템이 깊고 복잡할 수 있는 트리 구조에서 많은 다른 콘텐츠 아이템의 부모인 폴더인 경우 콘텐츠 아이템의 이동 또는 이름 변경은 오래된 위치 또는 경로로부터 새로운 위치 또는 경로로 모든 자손 콘텐츠 아이템을 삭제한 것으로 나타날 수도 있다.

[0214] 전술한 바와 같이, 콘텐츠 아이템이 새로운 위치에서 또는 새로운 이름으로 재 추가되기 전에 콘텐츠 아이템이 로컬 트리로부터 삭제 또는 제거되는 중간 상태는 바람직하지 않고, 이는 사용자의 데이터가 손실될 수 있는 데이터 취약성을 증가시킨다. 추가적으로 이동 또는 이름 변경 동작은 대응하는 추가 동작이 검출될 때까지 클라이언트 동기화 서비스에 삭제 동작으로 나타난다. 그러나 로컬 파일 시스템의 크기와 복잡성에 기초하여 삭제 동작이 검출된 후 추가 동작이 오랫동안 검출되지 않을 수 있다. 예를 들어, 클라이언트 동기화 서비스는 로컬 파일 시스템의 일 부분을 크롤링하며, 콘텐츠 아이템의 삭제를 발견하고, 콘텐츠 아이템이 로컬 파일 시스템의 다른 부분에 추가된 것을 발견하지 못하고, 이에 의해 클라이언트 동기화 서비스가 로컬 파일 시스템의 이 부분을 크롤링할 때까지 이동 동작을 완료할 수 있다.

[0215] 본 기술의 다양한 실시형태는 삭제 동작이 이동 또는 이름 변경 동작의 일부인지 또는 단순히 삭제 동작의 일부인지 여부를 결정하는 보다 효율적이고 빠른 방법을 제공함으로써 이들 및 다른 기술적 문제에 대한 기술적 해결책을 제공하는 것에 관한 것이다.

[0216] 도 15는 본 기술의 다양한 실시형태에 따른 이동 또는 이름 변경 동작에 응답하여 로컬 트리를 업데이트하는 예시적인 방법을 도시한다. 본 명세서에 기술된 방법 및 프로세스가 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(1500)은 예를 들어 클라이언트 디바이스에서 실행되는 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0217] 동작(1505)에서, 시스템은 콘텐츠 아이템에 대한 삭제 이벤트를 검출한다. 시스템은 클라이언트 디바이스의 로컬 파일 시스템 또는 시스템이 관리하도록 구성된 로컬 파일 시스템의 일부(예를 들어, 콘텐츠 관리 폴더)에 대한 변경 사항을 크롤링하거나 모니터링할 수 있다. 시스템은 로컬 파일 시스템과 로컬 트리 간의 차이를 식별하기 위해 로컬 파일 시스템과 로컬 트리를 비교할 수 있다. 삭제 이벤트는 하나 이상의 식별된 차이에 기초하여 검출될 수 있다. 예를 들어 콘텐츠 아이템의 노드는 특정 위치의 로컬 트리에 존재하지만 로컬 파일 시스템의 이 위치에 존재하는 것은 아니다. 이는 사용자나 애플리케이션이 이 위치로부터 콘텐츠 아이템이 제거되게 하여 시스템이 삭제 이벤트를 검출하게 하는 액션을 수행한 것을 나타낼 수 있다.

[0218] 삭제 이벤트를 검출하게 하는 액션은, 콘텐츠 아이템을 시스템에 의해 모니터링되는 다른 위치로 이동시키고, 콘텐츠 아이템을 시스템에 의해 모니터링되지 않는 다른 위치로 이동시키거나, (일부 파일 시스템에 의한 이동으로 처리될 수 있는) 콘텐츠 아이템의 이름을 변경하거나, 또는 실제로 콘텐츠 아이템을 삭제하는 사용자 또는 애플리케이션에 의해 야기될 수 있다. 삭제 이벤트를 야기한 사용자 액션이 무엇인지 및/또는 삭제 이벤트와 연

관된 추가 이벤트가 이미 검출되었는지 또는 검출되는지 여부를 결정하기 위해, 시스템은 동작(1510)에서 콘텐츠 아이템에 대한 동작 시스템으로 제공된 식별자를 식별할 수 있다.

[0219] 일부 실시형태에서, 동작 시스템으로 제공된 식별자는 아이노드(inode) 식별자일 수 있고, 콘텐츠 관리 시스템 및/또는 클라이언트 동기화 서비스에 의해 제공되는 파일 식별자와 상이하다. 많은 경우에, 동작 시스템은 무엇보다도 특히 아이노드 식별자에 기초하여 콘텐츠 아이템의 위치를 신속히 조회할 수 있기 위해 아이노드 식별자를 제공할 수 있다. 예를 들어, 일부 동작 시스템은 시스템이 아이노드 식별자를 키로 사용하여 콘텐츠 아이템의 현재 경로 또는 위치를 조회할 수 있는 인터페이스를 제공할 수 있다. 동작(1515)에서, 시스템은 콘텐츠 아이템의 위치에 대한 동작 시스템을 조회함으로써 콘텐츠 아이템의 위치를 결정할 수 있다. 조회에 응답하여, 동작 시스템은 아이노드 식별자에 의해 참조되는 콘텐츠 아이템의 로컬 파일 시스템에서 현재 위치 또는 경로로 리턴할 수 있다.

[0220] 콘텐츠 아이템의 현재 위치를 이용하여, 시스템은 삭제 이벤트를 야기한 액션이 무엇인지를 결정할 수 있다. 예를 들어, 현재 위치가 널(null) 위치이거나 콘텐츠 아이템이 더 이상 로컬 파일 시스템에 없다는 것을 나타내는 경우 삭제 이벤트를 야기한 액션은 실제 삭제이다. 따라서, 시스템은 로컬 트리로부터 콘텐츠 아이템에 대한 노드를 적절히 삭제할 수 있다. 현재 위치가 시스템(예를 들어, 클라이언트 동기화 서비스)에 의해 관리되지 않는 위치인 경우 삭제 이벤트를 야기한 액션은 콘텐츠 아이템을 이전 위치로부터 현재 위치로 이동시킨 것일 수 있다. 그러나 콘텐츠 아이템이 시스템에 의해 관리되는 영역 밖으로 이동되기 때문에 시스템은 더 이상 콘텐츠 아이템을 추적할 필요 없고 로컬 트리로부터 콘텐츠 아이템의 노드를 삭제할 수 있다.

[0221] 현재 위치가 시스템에 의해 여전히 관리되는 새로운 위치인 경우, 삭제 이벤트를 야기한 액션은 또한 콘텐츠 아이템을 이전 위치로부터 현재 위치로 이동시키는 것이다. 그러나 콘텐츠 아이템이 여전히 시스템에 의해 관리되는 영역 내에 있기 때문에 시스템은 대응하는 추가 이벤트의 검출을 대기하고, 삭제 이벤트와 추가 이벤트를 함께 이동 액션으로 처리하고, 로컬 트리를 원자 단위로 업데이트하여, 삭제 이벤트를 야기한 실제 액션을 미러 방식으로 반영한다.

[0222] 유사하게, 현재 위치가 시스템에 의해 관리되는 오래된 위치와 동일한 위치인 경우, 삭제 이벤트를 야기한 액션은 또한 이전 위치로부터 현재 위치로 콘텐츠 아이템의 이름을 변경하는 것이다. 일부 파일 시스템에서, 이름 변경 동작과 이동 동작은 이름 변경 동작이 하나의 이름을 가진 하나의 위치로부터 새로운 이름을 가진 동일한 위치로 이동하는 동작으로 처리된다는 것에서 관련된다. 따라서 시스템은 대응하는 추가 이벤트(새로운 이름을 가짐)의 검출을 대기하고, 삭제 이벤트와 추가 이벤트를 함께 이동 또는 이름 변경 액션으로 처리하고, 로컬 트리를 원자 단위로 업데이트하여, 삭제 이벤트를 야기한 실제 액션을 미러 방식으로 반영한다.

[0223] 따라서, 동작(1520)에서, 시스템은 삭제 이벤트가 콘텐츠 아이템의 위치에 기초하여 콘텐츠 아이템에 대한 추가 이벤트와 연관되어 있는지 여부를 결정한다. 삭제 이벤트가 추가 이벤트와 연관되어 있지 않은 경우, 삭제 이벤트는 동작(1525)에서 처리될 수 있다. 삭제 이벤트가 추가 이벤트와 연관되면, 시스템은 추가 이벤트를 기다릴 수 있고, 동작(1530)에서 콘텐츠 아이템에 대한 추가 이벤트를 검출할 수 있고, 동작(1535)에서 로컬 트리의 단일형 업데이트에서 추가 이벤트로 삭제 이벤트를 처리한다. 일부 구현예에 따르면, 아이노드 조회가 이미 콘텐츠 아이템에 대한 현재 위치 또는 경로를 제공했기 때문에 추가 이벤트를 기다리는 것은 불필요하다. 따라서, 시스템은 콘텐츠 아이템에 대한 경로를 관찰하고, 이 경로를 관찰된 경로 세트에 추가할 수 있다.

[0224] 도 14의 방법(1400)과 도 15의 방법(1500)은 별개로 설명되지만, 두 방법은 로컬 트리를 업데이트하기 위해 서로 연계하여 작동할 수 있다. 예를 들어, 삭제 이벤트가 추가 이벤트와 연관되어 있지 않은 경우, 삭제 이벤트는 도 15의 동작(1525)에서 삭제 이벤트를 대응하는 추가 이벤트와 결합하지 않고 처리될 수 있다. 일부 실시형태에 따르면, 삭제 이벤트를 처리하는 것은 도 14에 도시된 동작을 포함할 수 있고, 여기서, 예를 들어, 삭제 이벤트는 관찰된 경로 세트에 추가될 수 있고, 로컬 트리 제약이 위반되는지 여부를 결정하기 위해 검사될 수 있다.

[0225] 예를 들어, 삭제 이벤트와 연관된 콘텐츠 아이템이 로컬 트리에 하나 이상의 자손 노드를 갖는 경우, "비어 있지 않은 부모 노드는 삭제될 수 없다"는 제약이 위반될 수 있다. 이 위반에 대한 교정은 추가 경로(예를 들어, 콘텐츠 아이템의 모든 자손 노드에 대한 삭제 이벤트)를 관찰하려고 대기하는 것을 포함할 수 있다. 추가 파일 이벤트가 검출되면 추가 삭제 파일 이벤트가 추가적인 대응하는 추가 이벤트와 연관되어 있는지 여부를 결정하기 위한 검사를 포함하여 이러한 추가 파일 이벤트 또는 경로에 대한 제약이 검사될 수 있다. 일단 관찰된 모든 파일 이벤트의 유효성이 검증되면 파일 이벤트는 함께 일괄 배치(batched)되어 로컬 트리를 업데이트하는 데 사용될 수 있다.

[0226] 유사하게, 삭제 이벤트가 추가 이벤트와 연관된 경우, 시스템은 추가 이벤트를 기다릴 수 있고, 동작(1535)에서 로컬 트리에 대한 단일형 업데이트에서 추가 이벤트로 삭제 이벤트를 처리하는 것은 두 이벤트를 관찰된 파일 이벤트 세트에 추가하고, 임의의 로컬 트리 제약을 위반했는지 여부를 결정하고, 만약 위반된 경우 적절한 교정을 수행하고, 관찰된 전체 파일 이벤트 세트에 기초하여 로컬 트리를 업데이트하는 것을 포함할 수 있다.

[0227] 원격 트리 업데이트

[0228] 전술한 바와 같이, 원격 트리는 콘텐츠 관리 시스템에 의해 저장된 콘텐츠 아이템에 대한 서버 상태를 나타낸다. 예를 들어, 도 1의 서버 동기화 서비스(112)는 클라이언트 동기화 서비스(156)와 통신하며 클라이언트 디바이스(150)와 콘텐츠 관리 시스템(110) 사이의 콘텐츠 아이템에 대한 변경 사항을 동기화하도록 구성된다.

[0229] 본 기술의 다양한 실시형태는 콘텐츠 관리 시스템의 변경 사항에 기초하여 원격 트리를 업데이트하기 위한 다양한 기술적 해결책을 제공한다. 다른 트리 데이터 구조와 함께 원격 트리는 다양한 실시형태에서 클라이언트 디바이스와 콘텐츠 관리 시스템 사이의 동기화 프로세스에 중요하다. 예를 들어, 원격 트리에 대한 업데이트가 이루어지면 나머지 시스템은 업데이트에 반응하고, 일부 경우 원격 트리에 대한 변경 사항이 동기화되어 클라이언트 디바이스의 파일 시스템 상태에 적용될 수 있다. 따라서 원격 트리가 업데이트되는 방식에 주목해야 하는 것이 중요하다.

[0230] 전체적으로 더 상세히 설명되는 바와 같이, 특정 실시형태에서, 콘텐츠 관리 시스템(110)은 또한 변경, 액세스 등에 관한 데이터의 로그를 서버 파일 저널(148)에 저장할 수 있다. 서버 파일 저널(148)은 콘텐츠 아이템에 대한 리비전의 하나 이상의 저널을 콘텐츠 저장소(142)에 유지할 수 있다. 하나 이상의 저널은 각각의 네임스페이스에서 각 콘텐츠 아이템의 리비전을 추적할 수 있다. 서버 파일 저널(148)에서 저널의 값의 행은 네임스페이스에서 콘텐츠 아이템을 식별하고, 네임스페이스에서 콘텐츠 아이템의 상태를 반영할 수 있다. 네임스페이스에서 동일한 콘텐츠 아이템에 대응하는 저널의 후속 행은 네임스페이스에서 콘텐츠 아이템에 대한 후속 리비전을 반영할 수 있다.

[0231] 따라서, 콘텐츠 아이템과 연관된 서버 파일 저널(148)의 행은 콘텐츠 아이템의 현재 상태와, 생성으로부터 현재 상태로 콘텐츠 아이템에 대한 임의의 리비전을 식별할 수 있다. 콘텐츠 아이템 정보를 서버 파일 저널(148)과 동기화하기 위해, 콘텐츠 관리 시스템(110)은 서버 파일 저널(148)에 포함된 정보를, 클라이언트 디바이스(150)에 제공되어 클라이언트 디바이스(150)에 서버 파일 저널(148)로부터 콘텐츠 아이템의 최신 서버 상태를 제공할 수 있는 동작 데이터로 변환할 수 있다.

[0232] 본 기술의 다양한 실시형태는 콘텐츠 관리 시스템(110)으로부터 동작 데이터를 수신하고 동작 데이터에 기초하여 콘텐츠 관리 시스템에 저장된 콘텐츠 아이템에 대한 서버 상태를 나타내는 원격 트리를 업데이트하는 클라이언트 디바이스(150)에 관한 것이다. 그러나, 클라이언트 디바이스(150)에 제공된 동작 데이터는 원격 트리와 같은 트리 데이터 구조에 있지 않을 수 있다. 대신 동작 데이터는 동작 로그를 나타낸다. 따라서, 클라이언트 디바이스(150)에서 실행되는 클라이언트 동기화 서비스(156)는 동작 로그를 포함하는 동작 데이터를 수신하고 원격 트리에서 동작 로그를 실행하여 원격 트리를 업데이트하도록 구성된다.

[0233] 일부 실시형태에 따르면, 콘텐츠 관리 시스템(110)은 전체 원격 트리를 재구축하도록 구성된 동작 데이터를 생성 및 제공할 수 있다. 이것은 하나 이상의 네임스페이스에 대한 전체 동작 로그를 포함할 수 있다. 일부 경우에, 콘텐츠 관리 시스템(110)은 더 이상 현재 상태가 아니거나 원격 트리를 구축하는데 필요하지 않은 동작을 로그로부터 제거할 수 있다. 예를 들어, 이후에 삭제되는 콘텐츠 아이템에 대한 동작은 동작 데이터로부터 제거될 수 있다. 클라이언트 동기화 서비스(156)는 동작 로그를 수신하고, 로그에서 각 동작을 증분적으로 단계적으로 수행하여 완전한 원격 트리를 구축할 수 있다. 대안적으로, 클라이언트 동기화 서비스는 현존하는 원격 트리를 동작 로그와 비교하여, 원격 트리를 최신 상태로 만들기 위해 현존하는 원격 트리에 적용할 필요가 있는 동작을 로그에서 결정할 수 있다.

[0234] 다른 실시형태에서, 콘텐츠 관리 시스템(110)은 클라이언트 디바이스에 의해 저장된 원격 트리를 증분적으로 업데이트하도록 구성된 동작 데이터를 생성 및 제공할 수 있다. 동작 데이터를 생성하는 데 사용되는 로그 부분을 결정하기 위해, 콘텐츠 관리 시스템(110)은 네임스페이스에 대한 타임 라인의 포인트를 나타내는 커서를 사용한다. 커서는 예를 들어 특정 네임스페이스에 대응하는 서버 파일 저널(148)의 동작 로그에 엔트리 식별자를 포함할 수 있다. 일 실시형태에서, 엔트리 식별자는 네임스페이스에 대한 동작 로그에서 엔트리당 증가하는 SJ_ID일 수 있다. 그러나, 커서는 또한 서버 상태의 수명 사이클에서 포인트를 표시할 수 있는 논리적 클록 값, 카운터,

타임스탬프, 또는 임의의 다른 값으로 구현될 수 있다.

[0235] 예를 들어, 콘텐츠 관리 시스템(110)은 네임스페이스에 변경 사항이 있었다고 결정하고, 클라이언트 동기화 서비스(156)에 통지를 전송할 수 있다. 변경 사항의 통지를 수신한 것에 응답하여, 클라이언트 동기화 서비스(156)는 원격 트리가 마지막으로 업데이트된 이후 동작 로그 요청을 전송할 수 있다. 요청은 원격 트리가 마지막으로 업데이트된 시간, 또는 콘텐츠 관리 시스템(110)으로부터 수신된 마지막 업데이트를 나타내는 커서를 포함할 수 있다. 대안적으로, 클라이언트 동기화 서비스(156)는 콘텐츠 관리 시스템(110)으로부터 통지할 필요없이 커서를 포함하는 요청을 전송할 수 있다. 또 다른 구현예에서, 콘텐츠 관리 시스템(110)은 동작 데이터가 클라이언트 동기화 서비스(156)로 전송될 때마다 커서를 추적할 수 있어서, 클라이언트 동기화 서비스(156)가 커서를 콘텐츠 관리 시스템으로 전송할 필요가 없다.

[0236] 이 커서를 사용하여, 콘텐츠 관리 시스템(110)은 클라이언트 동기화 서비스(156)에 전송할 동작 로그 부분을 결정하고, 이 부분을 동작 데이터로 전송할 수 있다. 클라이언트 동기화 서비스(156)는 동작 데이터로서 동작 로그의 일부를 수신하고, 원격 트리를 업데이트하기 위해 로그에서 각 동작을 증분적으로 단계적으로 수행할 수 있다.

[0237] 그러나, 특정 기능을 구현하기 위해, 서버 상태와 이 서버 상태를 나타내는 원격 트리는 하나를 초과하는 네임스페이스를 포함할 수 있다. 예를 들어, 다수의 네임스페이스를 가지면 보다 조직 중심적인 저장 모델 및/또는 개인 및 그룹 사이의 공유를 가능할 수 있다. 도 16은 다양한 실시형태에 따른 트리 데이터 구조의 일례를 도시한다. 원격 트리(1600)가 도 16에 도시되어 있지만, 다른 트리 데이터 구조(예를 들어, 동기 트리 및 로컬 트리)도 유사한 구조 및 특성을 가질 수 있다. 원격 트리는 4개의 네임스페이스(1605, 1610, 1615 및 1620)를 포함할 수 있다. 네임스페이스(1605)는 루트 네임스페이스를 나타낼 수 있는 반면, 네임스페이스(1610 및 1615)는 네임스페이스(1605) 내에 마운트된다. 내포된 네임스페이스는 또한 도시된 바와 같이 네임스페이스(1610) 내 네임스페이스(1620)를 마운트하는 것에 의해 가능하다.

[0238] 각 네임스페이스는 하나 이상의 개별 사용자 및 상이한 권한과 연관될 수 있다. 예를 들어, 기업 네임스페이스(1605)는 회사 또는 조직과 전체적으로 연관될 수 있는 반면, 네임스페이스(1615)는 조직의 회계 부서와 연관될 수 있고, 네임스페이스(1610)는 조직 내의 엔지니어링 부서와 연관될 수 있다. 네임스페이스(1620)는 엔지니어링 부서 내의 그룹과 연관될 수 있고, 개별 사용자를 위한 네임스페이스를 포함하는 추가 네임스페이스를 포함할 수 있다. 상이한 네임스페이스를 사용하면 사용자 간에 콘텐츠 아이템을 공유하고 액세스하는 데 보다 나은 협력 및 제어를 할 수 있다.

[0239] 각각의 네임스페이스는 네임스페이스 식별자(예를 들어, NS_ID) 및 이 네임스페이스에 대한 타임 라인의 포인트를 나타내는 커서(예를 들어, SJ_ID)에 의해 식별된 별도의 동작 로그와 연관될 수 있다. 그러나 다수의 네임스페이스에 대해 SJ_ID를 사용하면 다수의 네임스페이스에 걸쳐 진행을 추적하고 타임 라인을 동기화하는 것이 어렵다. 예를 들어, 제1 네임스페이스에 대한 제1 SJ_ID가 제2 네임스페이스에 대한 제2 SJ_ID와 동일하면 제1 및 제2 네임스페이스가 동일한 상태 또는 특정 시점에 대응한다는 것을 보장할 수 없다.

[0240] 이것은 네임스페이스에 걸쳐 동작이 있을 때 심각한 기술적 문제를 나타낸다. 예를 들어 하나의 네임스페이스를 다른 네임스페이스로 마운트하는 동작은 두 네임스페이스의 동작 로그들 간에 의존성을 도입한다. 이동 동작과 같이 네임스페이스에 걸친 동작은 트리 데이터 구조에 둔 제약을 위반할 수 있다. 예를 들어, 네임스페이스(1620)의 콘텐츠 아이템(1650)을 네임스페이스(1615)로 이동시키는 도 16에 도시된 이동 동작(1655)은 네임스페이스(1620)에 대한 동작 로그에서 동작(예를 들어, 삭제 동작), 및 네임스페이스(1615)에 대한 동작 로그에서 대응하는 동작(예를 들어, 추가 동작)으로 나타날 수 있다. 파일 식별자가 트리에서 둘 이상의 위치에 존재할 수 없다는 제약을 유지하려면 네임스페이스(1620)에서 삭제 동작이 네임스페이스(1615)에서 추가 동작 전에 일어나야 한다. 그러나 각 네임스페이스의 로그에 대해 SJ_ID만을 사용하면 이를 보장하는 것은 어렵다.

[0241] 이하의 구획에서 더 상세하게 논의되는 바와 같이, 본 기술의 다양한 실시형태는 램포트 클록을 사용하여 다수의 네임스페이스의 다수의 로그 엔트리 식별자(예를 들어, SJ_ID)를 동기화하여 다수의 네임스페이스의 SJ_ID들 간의 순서 제약을 인코딩하고, 그 결과 네임스페이스에 걸쳐 전체 정렬하도록 구성된 콘텐츠 관리 시스템을 사용하는 기술적 해결책을 제공한다. 콘텐츠 관리 시스템은 각각의 네임스페이스에 대한 동작 로그를 선형화된 동작 세트로 선형화하도록 더 구성될 수 있으며, 이 선형화된 동작 세트는 동작 데이터에 포함되어 클라이언트 디바이스에 제공된다.

[0242] 클라이언트 디바이스와 관련하여, 클라이언트 동기화 서비스는 다수의 네임스페이스에 걸쳐 선형화되고 올바른

순서로 정렬된 동작 세트를 포함하는 동작 데이터를 수신할 수 있다. 클라이언트 동기화 서비스는 원격 트리를 업데이트하기 위해 선형화된 동작 세트 내의 각 동작을 커서를 사용하여 증분적으로 단계적으로 수행할 수 있다.

[0243] 원격 트리에서 네임스페이스의 마운트

[0244] 전술한 바와 같이, 클라이언트 동기화 서비스는 콘텐츠 관리 시스템으로부터 수신된 동작 데이터에 기초하여 클라이언트 디바이스 상의 원격 트리를 업데이트할 수 있다. 클라이언트 동기화 서비스가 동작 데이터에서 이전에 알려지지 않은 네임스페이스 타깃에 대한 마운트 동작을 만나면 추가 기술적 문제가 발생한다. 클라이언트 동기화 서비스는, 마운트 타깃의 콘텐츠를 결정하고, 원격 트리의 무결성을 보존하기 위해 다양한 제약 또는 규칙에 대해 이 콘텐츠의 유효성을 검증하기 전에 네임스페이스 타깃을 마운트하는 것이 방지된다. 그러나 마운트 타깃을 이전에는 알 수 없었으므로 클라이언트 동기화 서비스는 마운트 타깃의 콘텐츠를 인식하지 못한다. 본 기술의 다양한 실시형태는 이들 및 다른 기술적 문제를 다룬다.

[0245] 도 17은 다양한 실시형태에 따른 네임스페이스를 마운트하는 개념도를 도시한다. 콘텐츠 관리 시스템은 각 네임스페이스에 대한 동작 로그를 클라이언트 디바이스에 제공되는 선형화된 동작 세트로 선형화하도록 구성된다. 도 17은 네임스페이스(NSID1)(1710)에 대한 동작 로그 및 네임스페이스(NSID2)(1715)에 대한 동작 로그의 표현을 포함한다. 두 네임스페이스 사이의 네임스페이스 간 순서는 엔트리 로그 식별자(예를 들어, SJ_ID) 및 램포트 클록 값을 사용하여 수립된다.

[0246] 초기 시간 기간 동안, 콘텐츠 관리 시스템은 사용자 계정과 연관된 2개의 네임스페이스를 선형화하고 있다. 이벤트(1720)에서 네임스페이스(NSID2)(1715)에 대한 동작 로그는 SJ_ID 9, 클록 15에서 처리되어 새로운 네임스페이스가 네임스페이스(NSID2)(1715) 내에 마운트되는 것을 나타낸다. 이 시점까지 클라이언트 디바이스에는 새로운 네임스페이스와 연관된 정보가 없을 수 있고, 그리하여 네임스페이스가 즉시 마운트된 경우 트리 제약이 위반되지 않는다는 보장이 없다.

[0247] 콘텐츠 관리 시스템은 타깃 네임스페이스(1725)를 마운트하기 위한 마운트 동작(1720)을 검출한다. 마운트 동작(1720)을 검출하는 것에 응답하여, 콘텐츠 관리 시스템은 타깃 네임스페이스에 대한 네임스페이스 식별자를 포함하는 마운트 통지를 클라이언트 디바이스에 전송한다. 콘텐츠 관리 시스템은 또한 네임스페이스(NSID1)(1710) 및 네임스페이스(NSID2)(1715)와 함께 타깃 네임스페이스(1725)를 선형화 프로세스에 추가하고, 타깃 네임스페이스(1725)에 대한 동작 로그의 접두사(prefix)를 클라이언트 디바이스에 전송한다. 접두사는 마운트 동작(1720)까지 추가될 수 있는 타깃 네임스페이스(1725)에 대한 동작 로그의 초기 부분일 수 있고, 네임스페이스를 원격 트리에 마운트하기 전에 타깃 네임스페이스(1725)에 대한 서브 트리를 구축하기 위해 클라이언트 디바이스에 의해 사용될 수 있다.

[0248] 도 18은 본 기술의 다양한 실시형태에 따라 원격 트리에서 네임스페이스를 마운트하기 위한 예시적인 방법을 도시한다. 본 명세서에 기술된 방법 및 프로세스가 특정 순서로 특정 단계 및 동작으로 도시되었을 수 있지만, 달리 언급되지 않는 한, 유사하거나 대안적인 순서로 또는 병렬로 수행되는 추가적인, 더 적은, 또는 대안적인 단계 및 동작은 다양한 실시형태의 범위 내에 있다. 방법(1800)은 예를 들어 클라이언트 디바이스에서 실행되는 도 2의 클라이언트 동기화 서비스(156)와 같은 시스템에 의해 구현될 수 있다.

[0249] 동작(1805)에서, 시스템은 타깃 네임스페이스에 대한 마운트 통지를 수신할 수 있다. 마운트 통지는 콘텐츠 관리 시스템에 의해 시스템으로 전송되어 서브 트리가 타깃 네임스페이스에 대한 동작 로그의 수신 접두사 또는 초기 부분에 기초하여 생성될 것임을 시스템에 통지할 수 있다. 동작(1810)에서, 시스템은 타깃 네임스페이스에 대한 동작 로그의 초기 부분을 수신할 수 있다.

[0250] 도 17에 도시된 바와 같이, 타깃 네임스페이스에 대한 동작 로그의 접두사 또는 초기 부분(1730)은 콘텐츠 관리 시스템으로부터 시스템(예를 들어, 클라이언트 디바이스)으로 전송된다. 동작(1815)에서, 시스템은 동작 로그의 초기 부분에 기초하여 타깃 네임스페이스에 대한 서브 트리를 구축하기 시작할 수 있다. 도 17에 도시된 바와 같이, 타깃 네임스페이스(1750)에 대한 서브 트리는 시스템이 서브 트리의 구축을 완료할 때까지 선-원격(pre-remote) 유지 영역(1760)에서 생성될 수 있다.

[0251] 선-원격 유지 영역은 클라이언트 디바이스가 마운트 타깃(예를 들어, 마운트될 네임스페이스)과 연관된 하나 이상의 서브 트리를 저장 및 생성하기 위한 위치이다. 이러한 마운트 타깃에 대한 서브 트리는 서브 트리가 원격 트리의 현재 상태(예를 들어, 커서)에 도달하기 전까지 선-원격 유지 영역에 저장 및 업데이트될 수 있다. 마운트 타깃의 서브 트리가 원격 트리의 현재 상태에 도달하면, 마운트 타깃의 서브 트리가 원격 트리에 마운트될

수 있다.

[0252] 전술한 바와 같이, 추가적인 네임스페이스는 타깃 네임스페이스 내에 내포될 수 있고, 클라이언트 디바이스에 이전에 알려지지 않았을 수 있다. 동작 로그의 초기 부분(1730)은 처리되어 클라이언트 디바이스로 전송될 때, 추가 마운트가 발견될 수 있고, 방법(1800)이 재귀적으로 반복될 수 있다. 예를 들어, 타깃 네임스페이스에 대한 동작 로그의 초기 부분(1730)이 시스템으로 전송될 때, 콘텐츠 관리 시스템에 의해 추가 마운트 동작이 검출되고 이에 의해 콘텐츠 관리 시스템이 다른 타깃 네임스페이스에 대한 다른 마운트 통지를 전송하여, 방법(1800)의 제1 반복 내에서 방법(1800)의 재귀 반복을 개시할 수 있다. 방법(1800)의 재귀 반복이 (발견된 추가로 내포된 네임스페이스에 대한 추가 반복과 함께) 완료되면 프로세스는 방법(1800)의 제1 반복으로 돌아갈 수 있다.

[0253] 타깃 네임스페이스에 대한 동작 로그의 초기 부분(1730)이 클라이언트 디바이스로 전송되면, 콘텐츠 관리 시스템은 처음에 프로세스를 시작한 검출을 갖는 도 17의 마운트 동작(1720)을 전송할 수 있다. 클라이언트 디바이스는 마운트 동작을 수신하고, 동작(1820)에서, 원격 트리의 마운트 위치에서 타깃 네임스페이스에 대한 서브 트리를 마운트한다. 일부 실시형태에 따르면, 마운트 위치는 마운트 동작, 마운트 통지 또는 이 둘 모두에 의해 제공될 수 있다. 도 17에 도시된 바와 같이, 타깃 네임스페이스(1750)에 대한 서브 트리는 원격 트리(1780)의 마운트 위치(1770)에 마운트된다. 일부 실시형태에 따르면, 마운트된 서브 트리를 포함하는 원격 트리가 원격 트리에 적용된 모든 트리 제약과 일치하는 것을 보장하기 위해 추가적인 검증 검사가 수행될 수 있다.

[0254] 일부 실시형태에 따르면, 더 이상 유효하지 않거나, 최신 상태가 아니거나, 최종 서브 트리에 반영되지 않는 동작을 제거하기 위해 타깃 네임스페이스에 대한 동작 로그의 접두사 또는 초기 부분이 처리될 수 있다. 예를 들어, 초기 부분은 최종 서브 트리에 반영되지 않은 콘텐츠 아이템에 대한 삭제 동작, 최종 서브 트리에 반영되지 않은 콘텐츠 아이템에 대한 추가 및 대응하는 삭제 동작, 또는 타깃 네임스페이스 외부로 콘텐츠 아이템을 이동시키는 이동 동작을 포함할 수 있다. 이러한 유형의 동작은 타깃 네임스페이스의 최종 서브 트리에 반영되지 않고, 공간, 대역폭, 처리 시간 및 다른 컴퓨팅 자원을 줄이기 위해 동작 로그의 초기 부분으로부터 제거될 수 있다.

[0255] 콘텐츠 관리 시스템 파일 저널 및 저장 시스템

[0256] 콘텐츠 관리 시스템(110)으로 초점을 맞추면, 도 19a는 시스템 구성(100)에서 콘텐츠 관리 시스템(110)과 클라이언트 디바이스(150) 사이에서 콘텐츠를 동기화하기 위한 예시적인 아키텍처의 개략도를 도시한다. 이 예에서, 클라이언트 디바이스(150)는 콘텐츠 저장 인터페이스(1906) 및 파일 저널 인터페이스(1902)를 통해 각각 콘텐츠 저장소(142) 및 서버 파일 저널(148)과 상호 작용한다. 콘텐츠 저장 인터페이스(1906)는 콘텐츠 저장 서비스(116)에 의해 제공되거나 관리될 수 있고, 파일 저널 인터페이스(1902)는 서버 동기화 서비스(112)에 의해 제공되거나 관리될 수 있다. 예를 들어, 콘텐츠 저장 인터페이스(1906)는 콘텐츠 저장 서비스(116)의 서브 구성 요소 또는 서브 서비스일 수 있고, 파일 저널 인터페이스(1902)는 서버 동기화 서비스(112)의 서브 구성 요소 또는 서브 서비스일 수 있다.

[0257] 콘텐츠 저장 인터페이스(1906)는 클라이언트 디바이스(150)와 콘텐츠 저장소(142) 사이에 콘텐츠 요청 또는 상호 작용과 같은 통신을 관리할 수 있다. 콘텐츠 저장 인터페이스(1906)는 클라이언트 디바이스(150)로부터의 요청을 처리하여 콘텐츠를 콘텐츠 저장소(142)로 업로드하고 콘텐츠 저장소로부터 콘텐츠를 다운로드할 수 있다. 콘텐츠 저장 인터페이스(1906)는 클라이언트 디바이스(150)로부터 콘텐츠 요청(예를 들어, 다운로드, 업로드 등)을 수신하고, 액세스 제어 리스트(145)에서 권한을 검증하고, 권한 부여 서비스(132)와 통신하며 클라이언트 디바이스(150)(및/또는 클라이언트로부터의 요청)가 콘텐츠를 콘텐츠 저장소(142)로 업로드하거나 콘텐츠 저장소로부터 콘텐츠를 다운로드할 권한이 부여되었는지 여부를 결정하고, 콘텐츠 저장소(142)와 상호 작용하며 콘텐츠 저장소(142)의 콘텐츠를 클라이언트 디바이스(150)로 다운로드하거나 콘텐츠를 업로드할 수 있다. 디바이스(150)로부터의 요청이 콘텐츠 아이템을 다운로드하는 요청이라면 콘텐츠 저장 인터페이스(1906)는 콘텐츠 저장소(142)로부터 콘텐츠 아이템을 검색하고, 콘텐츠 아이템을 클라이언트 디바이스(150)에 제공할 수 있다. 클라이언트 디바이스(150)로부터의 요청이 콘텐츠 아이템을 업로드하는 요청이라면, 콘텐츠 저장 인터페이스(1906)는 클라이언트 디바이스(150)로부터 콘텐츠 아이템을 획득하고, 저장을 위해 콘텐츠 아이템을 콘텐츠 저장소(142)에 업로드할 수 있다.

[0258] 클라이언트 디바이스(150)로부터 콘텐츠 요청을 처리할 때, 콘텐츠 저장 인터페이스(1906)는 저장 인덱스(1910)와 통신하며 콘텐츠 저장소(142)에서 요청된 콘텐츠의 이용 가능성 및/또는 저장 위치를 검사하고, 콘텐츠 저장소(142)에서 콘텐츠 아이템을 추적할 수 있다. 저장 인덱스(1910)는, 콘텐츠 저장소(142) 상의 콘텐츠 아이템

을 식별하고 또한 콘텐츠 저장소(142) 내의 콘텐츠 아이템의 각각의 위치를 식별할 수 있는 콘텐츠 아이템의 인덱스를 콘텐츠 저장소(142) 상에 유지할 수 있다. 따라서, 저장 인덱스(1910)는 콘텐츠 저장소(142) 상의 콘텐츠 아이템뿐만 아니라 콘텐츠 아이템의 저장 위치를 추적할 수 있다. 저장 인덱스(1910)는 파일과 같은 전체 콘텐츠 아이템, 및/또는 블록 또는 청크와 같은 콘텐츠 아이템의 일부를 추적할 수 있다. 일부 경우에, 콘텐츠 아이템은, 콘텐츠 저장소(142)에 저장될 수 있고 저장 인덱스(1910)에서 추적될 수 있는 블록 또는 청크로 분할될 수 있다. 예를 들어, 콘텐츠 저장소(142)는 콘텐츠 아이템의 각각의 데이터 부분을 포함하는 데이터의 블록 또는 청크로서 콘텐츠 아이템을 저장할 수 있다. 저장 인덱스(1910)는 콘텐츠 저장소(142)에 저장된 콘텐츠 아이템의 블록 또는 청크를 추적할 수 있다. 후술되는 도 19b는 콘텐츠 아이템의 블록을 저장 및 추적하기 위한 예시적인 구성을 도시한다.

[0259] 파일 저널 인터페이스(1902)는 클라이언트 디바이스(150)와 서버 파일 저널(148) 사이에 메타데이터 요청, 및 콘텐츠 동기화 및 동작과 같은 통신을 관리할 수 있다. 예를 들어, 파일 저널 인터페이스(1902)는 클라이언트 디바이스(150)와 서버 파일 저널(148) 사이에 동작, 구성, 및 상태 정보를 변환, 검증, 인증 및/또는 처리할 수 있다. 파일 저널 인터페이스(1902)는 커서 내의 FSAuth 토큰으로부터 또는 권한 부여 서비스(132)를 통해 권한을 검증하여 클라이언트 디바이스(150)에 의해 서버 파일 저널(148)로 전송된 요청을 허가하거나 또는 요청의 허가를 검증할 수 있다. 클라이언트 디바이스(150)로부터의 요청 또는 동작을 처리할 때, 파일 저널 인터페이스(1902)는 네임스페이스 멤버십 저장소(1908)에 액세스하여 클라이언트 디바이스(150)로부터의 요청 또는 동작과 연관된 네임스페이스에 대한 네임스페이스 소유권 정보를 결정 또는 검증하고, 액세스 제어 리스트(145)로부터 권한 정보를 검색하여 클라이언트 디바이스(150)로부터의 요청 또는 동작과 연관된 콘텐츠의 권한을 검증할 수 있다.

[0260] 파일 저널 인터페이스(1902) 내 변환 서비스(1904)는 클라이언트 디바이스(150)와 서버 파일 저널(148) 사이의 통신을 위한 선형화 및 변환 동작을 수행할 수 있다. 예를 들어, 변환 서비스(1904)는 클라이언트 디바이스(150)로부터의 통신을 서버 파일 저널(148) 내의 데이터의 구조 및 포맷과 일치하는 다른 포맷으로 변환할 수 있고, 그 역도 마찬가지이다. 예시를 위해, 일부 경우에, 클라이언트 디바이스(150)는 클라이언트 디바이스(150)에서 콘텐츠 아이템 정보(예를 들어, 상태, 변경, 버전 등)를 동작으로서 처리할 수 있는 반면, 서버 파일 저널(148)은 데이터베이스 테이블과 같은 데이터 구조의 행으로 반영된 콘텐츠 아이템 리비전과 동일한 정보를 처리할 수 있다. 클라이언트 디바이스(150)와 서버 파일 저널(148) 사이의 콘텐츠 아이템 정보의 동기화를 구현하기 위해, 변환 서비스(1904)는 클라이언트 디바이스(150)로부터의 동작을 서버 파일 저널(148)에 적합한 리비전으로 변환할 수 있고, 서버 파일 저널(148) 상의 데이터의 행에 반영된 리비전을 클라이언트 디바이스(150)에 적합한 동작에 변환할 수 있다.

[0261] 일부 경우에, 권한 부여 서비스(132)는 클라이언트 디바이스(150)가 요청된 콘텐츠 아이템에 액세스, 요청된 콘텐츠 아이템을 업데이트, 다운로드 또는 업로드하도록 허가되었는지를 확인 또는 나타내는 토큰을 생성할 수 있다. 토큰은 클라이언트 디바이스(150)와 연관된 디바이스 식별자, 클라이언트 디바이스(150)에서 인증되었거나 또는 허가된 사용자 계정과 연관된 계정 식별자, 클라이언트 디바이스(150)에서 허가된 세션과 연관된 세션 식별자, 뷰 컨텍스트 및 식별된 콜렉션에 대한 액세스 권한을 포함할 수 있다. 토큰은 커서라고 불리는 암호로 서명된 데이터 객체에 포함될 수 있으며, 이에 대해서는 아래에서 더 자세히 설명한다. 콘텐츠 관리 시스템(110) 및/또는 권한 부여 서비스(132)는 토큰(들)을 클라이언트 디바이스(150)에 전송할 수 있고, 클라이언트 디바이스(150)는 아래에 더 설명된 바와 같이 콘텐츠 아이템 리비전 및/또는 업데이트를 서버 파일 저널(148)에 요청할 때 콘텐츠 관리 시스템(110)에 토큰을 제공할 수 있다. 클라이언트 디바이스(150)는 또한 임의의 콘텐츠 요청(예를 들어, 다운로드, 업로드 등)을 검증하기 위해 콘텐츠 저장 인터페이스(1906)에 토큰을 제공할 수 있다. 콘텐츠 저장 인터페이스(1906)는 토큰을 사용하여 저장 인덱스(1910)를 조회하는 것을 허가하고 콘텐츠 아이템을 콘텐츠 저장소(142)에 업로드하거나 콘텐츠 저장소로부터 콘텐츠 아이템을 다운로드할 수 있다.

[0262] 예를 들어, 클라이언트 디바이스(150)는 콘텐츠 아이템을 콘텐츠 저장소(142)에 업로드하기 위한 요청을 콘텐츠 저장 인터페이스(1906)로 전송할 수 있다. 요청은 업로드될 콘텐츠 아이템 및 토큰을 포함할 수 있다. 콘텐츠 저장 인터페이스(1906)는 토큰을 사용하여 저장 인덱스(1910)를 조회하는 것을 허가하여, 콘텐츠 아이템이 콘텐츠 저장소(142) 상에 이미 존재하는지 여부를 검사하고 콘텐츠 아이템을 콘텐츠 저장소(142)로 업로드하도록 허가할 수 있다. 클라이언트 디바이스(150)는 또한 파일 저널 인터페이스(1902)에 토큰을 제공하여 서버 파일 저널(148)에 메타데이터를 저장하라는 요청을 허가하여 콘텐츠 아이템의 업로드 및 리비전을 추적할 수 있다.

[0263] 도 19b는 예시적인 블록 저장 및 동기화 구성을 도시한다. 이 예에서, 콘텐츠 저장소(142)는 특정 크기(예를 들어, 4MB)까지 콘텐츠 아이템(예를 들어, 파일)의 불투명 청크일 수 있는 데이터 블록을 저장할 수 있다. 콘텐츠

아이템은 블록으로 분할될 수 있고, 블록은 액세스를 위해 콘텐츠 저장소(142)에 저장될 수 있다. 저장 인덱스(1910)는 콘텐츠 저장소(142)에 저장된 블록뿐만 아니라 콘텐츠 저장소(142)에 저장된 블록의 각각의 위치를 추적할 수 있다. 파일 저널 인터페이스(1902)는 서버 파일 저널(148)과 상호 작용하며 콘텐츠 저장소(142)에 저장된 콘텐츠 아이템 및/또는 블록에 대한 리비전을 추적할 수 있다.

[0264] 예를 들어, 콘텐츠 아이템(1920)(예를 들어, MyFile.abc)은 블록(1920A, 1920B, 1920C, 1920N)으로 분할될 수 있다. 콘텐츠 저장 인터페이스(1906)는 블록(1920A, 1920B, 1920C, 1920N)을 수신하고, 콘텐츠 저장소(142)에 저장을 위해 블록 데이터(1922B)를 콘텐츠 저장소(142)로 전송할 수 있다. 블록 데이터(1922B)는 콘텐츠 아이템(1920)과 연관된 블록(1920A, 1920B, 1920C, 1920N)을 포함할 수 있다.

[0265] 블록(1920A, 1920B, 1920C, 1920N)은 콘텐츠 저장소(142)에서 하나 이상의 저장 디바이스 또는 볼륨에 저장되거나 및/또는 하나 이상의 논리적 저장 컨테이너(예를 들어, 버킷) 또는 데이터 클러스터 내에 집계될 수 있다. 일부 경우에, 블록(1920A, 1920B, 1920C, 1920N)은 동일한 위치(예를 들어, 저장 디바이스, 볼륨, 컨테이너 및/또는 클러스터)에 함께 저장될 수 있다. 다른 경우에, 블록(1920A, 1920B, 1920C, 1920N)의 일부 또는 전부는 2개 이상의 상이한 위치(예를 들어, 2개 이상의 상이한 저장 디바이스, 볼륨, 컨테이너 및/또는 클러스터)에 저장될 수 있다.

[0266] 콘텐츠 저장 인터페이스(1906)는 또한 저장 인덱스(1910)에 블록 메타데이터(1922A)를 저장할 수 있다. 블록 메타데이터(1922A)는 블록(1920A, 1920B, 1920C, 1920N)을 식별하고, 저장 인덱스(1910)가 콘텐츠 저장소(142)에서 블록(1920A, 1920B, 1920C, 1920N)을 추적할 수 있게 한다. 블록 메타데이터(1922A)는 각 블록(1920A, 1920B, 1920C, 1920N)에 대한 식별자를 포함할 수 있다. 블록에 대한 식별자는 블록을 식별하는 블록의 해시와 같은 이름 또는 키일 수 있다.

[0267] 블록 메타데이터(1922A)는 또한 블록(1920A, 1920B, 1920C, 1920N)의 각각의 저장 위치를 나타내는 블록(1920A, 1920B, 1920C, 1920N)에 대한 위치 정보를 포함할 수 있다. 블록의 위치 정보는 블록이 저장된 저장 디바이스 또는 볼륨, 및/또는 블록이 포함된 논리 저장 컨테이너 또는 데이터 클러스터를 식별할 수 있다. 위치 정보는 연관된 블록에 액세스하거나 연관된 블록을 검색하는 데 사용될 수 있다.

[0268] 콘텐츠 저장 인터페이스(1906)는 콘텐츠 저장소(142)에 블록(1920A, 1920B, 1920C, 1920N)을 저장하기 전에 또는 후에 저장 인덱스(1910)에 블록 메타데이터(1922A)를 저장할 수 있다. 예를 들어, 콘텐츠 저장 인터페이스(1906)는 블록(1920A, 1920B, 1920C, 1920N)이 콘텐츠 저장소(142)에 저장되었음을 나타내기 위해 블록(1920A, 1920B, 1920C, 1920N)을 콘텐츠 저장소(142)에 저장하고 이후 블록 메타데이터(1922A)를 저장 인덱스(1910)에 저장할 수 있다.

[0269] 일부 경우에, 콘텐츠 저장 인터페이스(1906)는 콘텐츠 저장소(142)에 블록(1920A, 1920B, 1920C, 1920N)을 저장하기 전에 저장 인덱스(1910)를 조회하여 블록(1920A, 1920B, 1920C, 1920N)이 콘텐츠 저장소(142)에 저장되어 있는지 여부를 결정할 수 있다. 예를 들어, 콘텐츠 저장 인터페이스(1906)는 블록 메타데이터(1922A)에 기초하여 저장 인덱스(1910)를 조회하여 블록(1920A, 1920B, 1920C, 1920N)이 콘텐츠 저장소(142)에 저장되어 있는지 여부를 검사할 수 있다. 저장 인덱스(1910)는 블록 메타데이터(1922A)의 블록 식별자를 저장 인덱스(1910)의 블록 식별자를 비교하여 일치하는지 여부를 검사할 수 있다. 블록 식별자들 사이에 일치하면 연관된 블록이 콘텐츠 저장소(142)에 저장되어 있는 것을 나타낸다.

[0270] 전술한 바와 같이, 서버 파일 저널(148)은 콘텐츠 아이템 추가, 편집, 이동 또는 이름 변경, 삭제 등을 포함하는 콘텐츠 아이템 리비전을 추적한다. 따라서, 파일 저널 인터페이스(1902)는 콘텐츠 아이템(1920) 및/또는 블록(1920A, 1920B, 1920C, 1920N)이 콘텐츠 저장소(142)에 추가되었다는 것을 나타내기 위해 서버 파일 저널(148)에 리비전(1922C)을 저장할 수 있다. 리비전(1922C)은 서버 파일 저널(148)에서 콘텐츠 아이템 리비전의 저널 내의 콘텐츠 아이템(1920)의 리비전을 나타낼 수 있다.

[0271] 리비전(1922C)은 콘텐츠 아이템(1920) 및 콘텐츠 아이템(1920)과 연관된 동작, 예를 들어 추가 동작(예를 들어, 업로드), 편집 동작, 이동 또는 이름 변경 동작, 삭제 동작 등과 같은 동작을 식별할 수 있다. 리비전(1922C)은 또한 콘텐츠 아이템(1920)이 저장된 콘텐츠 관리 시스템(110)에서의 네임스페이스를 식별할 수 있고, 리비전(1922C)을 저장하기 위한 서버 파일 저널(148)에서의 콘텐츠 아이템 리비전의 저널 내의 행을 식별할 수 있다. 콘텐츠 아이템 리비전의 저널에서의 행은 콘텐츠 아이템(1920)에 대한 리비전(1922C)과 연관된 리비전 번호를 나타낼 수 있다.

[0272] 파일 저널 인터페이스

[0273] 도 19c는 클라이언트 디바이스(150)와 서버 파일 저널(148) 사이의 파일 저널 인터페이스(1902)에 의해 처리되는 통신을 도시한다. 서버 파일 저널(148)은 콘텐츠 아이템 상태와 변경 사항(예를 들어, 리비전)을 서버 파일 저널(148)에서의 행과 필드의 값으로서 추적한다. 예를 들어, 서버 파일 저널(148)은 콘텐츠 저장소(142)에 콘텐츠 아이템에 대한 하나 이상의 리비전 저널을 유지할 수 있다. 하나 이상의 저널은 각각의 네임스페이스 상의 각 콘텐츠 아이템의 리비전을 추적할 수 있다. 서버 파일 저널(148) 상의 저널 내의 값의 행은 네임스페이스에서 콘텐츠 아이템을 식별할 수 있고, 네임스페이스에서 콘텐츠 아이템의 상태를 반영할 수 있다. 네임스페이스에서 동일한 콘텐츠 아이템에 대응하는 저널의 후속 행은 네임스페이스에서 콘텐츠 아이템에 대한 후속 리비전을 반영할 수 있다. 따라서, 콘텐츠 아이템과 연관된 서버 파일 저널(148)의 행은 콘텐츠 아이템의 현재 상태 및 생성으로부터 현재 상태까지 콘텐츠 아이템에 대한 임의의 리비전을 식별할 수 있다.

[0274] 콘텐츠 아이템 정보(예를 들어, 상태, 변경 또는 리비전 등)를 클라이언트 디바이스(150)와 동기화하기 위해, 서버 파일 저널(148)은 하나 이상의 콘텐츠 아이템에 대한 서버 파일 저널(148)에서 추적 또는 저장되는 리비전을 나타내는 파일 저널 인터페이스(1902)로 리비전 데이터(1934)를 송신하거나 또는 파일 저널 인터페이스로부터 리비전 데이터를 수신할 수 있다. 리비전 데이터(1934)는, 예를 들어, 서버 파일 저널(148)의 행에 대응하는 콘텐츠 아이템 리비전의 로그를 포함할 수 있다. 서버 파일 저널(148)은 리비전 데이터(1934)를 파일 저널 인터페이스(1904)로 전송할 수 있으며, 이 파일 저널 인터페이스는 아래에 더 설명되는 바와 같이 리비전 데이터(1934)를 클라이언트 디바이스(150)에 대한 동작 데이터(1932)로 변환할 수 있다.

[0275] 클라이언트 디바이스(150)는 클라이언트 디바이스(150)에서 콘텐츠 아이템을 업데이트 또는 수정하기 위해 콘텐츠 동작을 수행할 수 있다. 콘텐츠 아이템 정보를 서버 파일 저널(148)과 동기화하기 위해, 클라이언트 디바이스(150)는 파일 저널 인터페이스(1902)로 동작 데이터(1932)를 송신하거나 파일 저널 인터페이스로부터 동작 데이터를 수신할 수 있다. 클라이언트 디바이스(150)는 클라이언트 디바이스(150)에서 콘텐츠 아이템에 대한 변경을 보고하기 위해 동작 데이터(1932)를 파일 저널 인터페이스(1902)로 송신하고, 서버 파일 저널(148)로부터 콘텐츠 아이템의 최신 상태(예를 들어, 리비전 데이터(1934))를 얻기 위해 파일 저널 인터페이스(1902)로부터 동작 데이터(1932)를 수신할 수 있다.

[0276] 예를 들어, 클라이언트 디바이스(150)는 클라이언트 디바이스(150)에서 콘텐츠 아이템(A)을 편집할 수 있고, 콘텐츠 아이템(A)에 대한 편집을 나타내는 편집 동작을 파일 저널 인터페이스(1902)에 보고할 수 있다. 편집 동작은 콘텐츠 아이템(A)에 대한 리비전을 나타내기 위해 파일 저널 인터페이스(1902)와 통신되는 동작 데이터(1932)에 포함될 수 있다. 파일 저널 인터페이스(1902)는 편집 동작을 포함하는 동작 데이터(1932)를 수신하고, 서버 파일 저널(148)에 저장을 위해 리비전을 생성하여 콘텐츠 아이템(A)에 대한 편집을 추적할 수 있다. 파일 저널 인터페이스(1902)는 콘텐츠 아이템(A)의 편집된 상태를 나타내는 리비전을 저장하기 위해 서버 파일 저널(148)을 업데이트하기 위해, 서버 파일 저널(148)에 대한 리비전 데이터(1934)에서의 편집 동작과 연관된 리비전을 포함할 수 있다.

[0277] 이하에 더 설명되는 바와 같이, 동작 데이터(1932)는 클라이언트 디바이스(150)와 연관된 각각의 네임스페이스에 대해 클라이언트 디바이스(150)에 의해 획득된 최신 상태 또는 리비전을 식별하는 커서를 포함할 수 있다. 예를 들어, 커서는 클라이언트 디바이스(150)와 연관된 각각의 네임스페이스에 대해 클라이언트 디바이스(150)에 의해 획득된 서버 파일 저널(148)에서의 최신 리비전을 식별할 수 있다. 커서 내의 정보는 파일 저널 인터페이스(1902)가 클라이언트 디바이스(150)로부터의 동작 데이터(1932)에서의 동작이 동작과 연관된 네임스페이스(들)에 대한 서버 파일 저널(148)의 최신 상태 또는 리비전을 반영하는지 여부를 결정할 수 있게 한다. 이것은 파일 저널 인터페이스(1902)가 서버 파일 저널(148)의 더 오래된 리비전에 대응하는 클라이언트 디바이스(150)로부터의 동작 데이터(1932)에서의 동작이 서버 파일 저널(148)에 기입되지 않아서 서버 파일 저널(148)에 현존하는 리비전과 동작 데이터(1932)로부터 변환된 리비전 사이에 충돌이 야기되는 것을 방지하는 것을 보장하는 것을 도와줄 수 있다.

[0278] 클라이언트 디바이스(150)와 서버 파일 저널(148) 사이의 콘텐츠 아이템 정보의 동기화를 구현하기 위해, 파일 저널 인터페이스(1902)는 (예를 들어, 변환 서비스(1904)를 통해) 동작 데이터(1932)를 리비전 데이터(1934)로 변환할 수 있고, 그 역도 마찬가지이다. 클라이언트 디바이스(150)로부터 동작 데이터(1932)를 수신할 때, 파일 저널 인터페이스(1902)는 동작 데이터(1932)를 리비전 데이터(1934)로 변환할 수 있고, 리비전 데이터는 동작 데이터(1932)의 동작으로부터 해석된 콘텐츠 아이템 리비전을 포함한다. 서버 파일 저널(148)로부터 리비전 데이터(1934)를 수신할 때, 파일 저널 인터페이스(1902)는 리비전 데이터(1934)를 동작 데이터(1932)로 변환할 수 있고, 이 동작 데이터는 클라이언트 디바이스(150)에서 리비전 데이터(1934)의 리비전을 구현하기 위한 동작을 포함한다. 리비전 데이터(1934)는 하나 이상의 콘텐츠 아이템(즉, 하나 이상의 콘텐츠 아이템에 대한 리비전)에

일어난 것을 기술하는 서버 파일 저널(148)의 데이터를 포함하고, 동작 데이터(1932)는 하나 이상의 콘텐츠 아이템을 수정하기 위해 클라이언트 디바이스(150)에서 실행되었거나 실행되어야 하는 동작을 포함한다. 따라서, 파일 저널 인터페이스(1902)는 서버 파일 저널(148)(예를 들어, 동작 데이터(1934))로부터 하나 이상의 콘텐츠 아이템에 대한 리비전을 기술하는 데이터를, 클라이언트 디바이스(150)에서 하나 이상의 콘텐츠 아이템을 수정하기 위해 클라이언트 디바이스(150)에서 실행되었거나 실행되어야 하는 동작으로 변환할 수 있다.

[0279] 전술한 바와 같이, 클라이언트 디바이스(150)로부터 동작 데이터(1932)를 서버 파일 저널(148)에 대한 리비전 데이터(1934)로 변환하는 것에 더하여, 파일 저널 인터페이스(1902)는 서버 데이터 저널(148)로부터의 리비전 데이터(1934)를 클라이언트 디바이스(150)에 대한 동작 데이터(1932)로 변환할 수 있다. 파일 저널 인터페이스(1902)는 서버 파일 저널(148)로부터 리비전 데이터(1934)를 획득하고, 리비전 데이터(1934) 내의 리비전을, 이러한 리비전에 따라 클라이언트 디바이스(150)에서 하나 이상의 콘텐츠 아이템을 리비전하기 위해, 클라이언트 디바이스(150)에서 실행하기 위한 동작으로 변환할 수 있다. 리비전 데이터(1934) 내의 리비전으로부터 생성된 동작은 파일 저널 인터페이스(1902)에 의해 클라이언트 디바이스(150)로 제공되는 동작 데이터(1932)에 포함된다. 동작 데이터(1932)와 리비전 데이터(1934) 사이의 이러한 변환은 클라이언트 디바이스(150)와 서버 파일 저널(148)이 필요에 따라 서로 콘텐츠 아이템 정보를 동기화할 수 있게 한다.

[0280] 클라이언트 디바이스(150)에 의해 제공된 동작 데이터(1932)로부터 생성된 임의의 리비전 데이터(1934)를 서버 파일 저널(148)에 기입하기 전에, 파일 저널 인터페이스(1902)는 동작 데이터(1932)에서 커서를 검사하고 및/또는 서버 파일 저널(148)을 조회하여 리비전 데이터(1934) 내의 임의의 리비전이 서버 파일 저널(148)에서 충돌을 야기하지 않는 것을 보장할 수 있다. 예를 들어, 파일 저널 인터페이스(1902)는 서버 파일 저널(148)을 조회하여 리비전 데이터(1934) 내의 리비전과 연관된 콘텐츠 아이템의 버전이 서버 파일 저널(148)의 콘텐츠 아이템의 버전과 동일한지 여부를 검사할 수 있고, 또는 서버 파일 저널(148)에서의 콘텐츠 아이템의 버전이 리비전 데이터(1934) 내의 리비전이 속하는 콘텐츠 아이템과 업데이트된 버전 또는 다른 버전인지 여부를 검사할 수 있다. 서버 파일 저널(148)이 최신 버전의 콘텐츠 아이템이 리비전 데이터(1934)가 속하는 버전과 다른 버전인 것을 나타내면, 두 버전은 충돌한다.

[0281] 파일 저널 인터페이스(1902)는 서버 파일 저널(148)을 업데이트하여 동작 데이터(1932)로부터 도출된 리비전 데이터(1934)에 포함된 새로운 리비전을 저장할 수 있다. 서버 파일 저널(148)에서 리비전을 조회 및/또는 업데이트할 때, 파일 저널 인터페이스(1902)는 네임스페이스 멤버십 저장소(1908)를 조회하여, 리비전 데이터(1934) 내의 리비전에 의해 영향을 받는 임의의 네임스페이스와 연관된 네임스페이스 소유권 정보를 검색할 수 있다. 네임스페이스 소유권 정보는 어떤 사용자 계정(들)이 특정 네임스페이스의 구성원을 소유하거나 특정 네임스페이스의 구성원이고, 이에 따라 특정 네임스페이스에 액세스할 수 있는지를 나타낼 수 있다. 따라서, 파일 저널 인터페이스(1902)는 네임스페이스 소유권 정보를 분석하여, 서버 파일 저널(148)이 네임스페이스의 구성원이 아닌 사용자 계정으로부터 네임스페이스에 대한 리비전을 포함하도록 업데이트되지 않는 것을 보장할 수 있다.

[0282] 도 19d를 참조하면, 서버 파일 저널(148)은 콘텐츠 아이템 리비전 및 상태를 추적하고 식별하기 위해 저널(1960, 1962)을 저장할 수 있다. 이 예에서 저널(1960)은 네임스페이스 식별자(NS_ID), 서버 저널 식별자(SJ_ID), 경로, 블록, 이전 버전(Prev_Rev), 및 타깃 네임스페이스(타깃_NS)를 포함한 레코드를 포함한다. NS_ID는 서버 파일 저널(148)에서 네임스페이스를 고유하게 식별하기 위한 하나 이상의 값을 포함할 수 있다. SJ_ID는, 네임스페이스에 대해 주어진 저널의 행에 매핑되고 이 네임스페이스 내에서의 동작 또는 리비전의 순서를 제공하는 단조 증가하는 값을 포함할 수 있다. 경로는 연관된 콘텐츠 아이템을 식별하는 네임스페이스-상대 경로(namespace-relative path)일 수 있다. Prev_Rev는 경로와 연관된 콘텐츠 아이템의 이전 상태에 대응하는 행의 SJ_ID를 식별한다. 타깃_NS는 마운트된 네임스페이스의 마운트 지점에 대한 타깃 네임스페이스의 NS_ID를 식별한다. 타깃_NS 필드는 마운트 지점에 대응하지 않는 행(예를 들어, 리비전)에서는 설정되지 않는다.

[0283] 저널(1962)은 NS_ID, SJ_ID, 클록(예를 들어, 타임스탬프), 파일 식별자(File_ID), 확장된 속성(xattr) 등을 포함하는 레코드를 포함한다. xattr은 콘텐츠 아이템 또는 동작과 연관된 메타데이터를 저장할 수 있다.

[0284] 일부 경우에, 저널(1960)은 연관된 콘텐츠 아이템의 크기를 나타내는 크기 필드; 콘텐츠 아이템이 디렉토리일 때 나타내도록 설정될 수 있는 디렉토리 필드(예를 들어, Is_Dir); 연관된 파일을 고유하게 식별하는 파일 식별자; 클록 또는 타임스탬프 필드; 등과 같은 다른 필드를 포함할 수 있다.

[0285] 파일 저널 인터페이스(1902)는 전술한 바와 같이 동작 데이터(1932)와 리비전 데이터(1934)에 기초하여 변환(1970)을 수행할 수 있다. 변환(1970)을 수행할 때, 변환 서비스(1904)는 동작 데이터(1932)를 리비전(1972)으로 변환할 수 있고, 이 리비전은 서버 파일 저널(148)에 저장을 위해 선형화된 리비전을 포함한다. 변환 서비스

(1904)는 리비전 데이터(1934)를 클라이언트 디바이스(150)로 전송된 동작 데이터(1932)에 포함된 선형화된 동작(1974A)으로 변환할 수 있고, 이 선형화된 동작은 클라이언트 디바이스(150)에서 콘텐츠 아이템 정보(예를 들어, 상태, 변경 등)를 업데이트하기 위해 클라이언트 디바이스(150)에 의해 적용될 수 있다. 변환 서비스(1904)는 또한 커서(1974B)를 생성 또는 업데이트하고, 동작 데이터(1932)에서 커서(1974B)를 클라이언트 디바이스(150)에 제공할 수 있다. 커서(1974B)는 선형화된 동작(1974B)과 연관된 각각의 네임스페이스 및/또는 콘텐츠 아이템에 대응하는 서버 파일 저널(148)에서 각각의 리비전 또는 행을 식별한다.

[0286] 예를 들어, 커서(1974B)는 네임스페이스(예를 들어, NS_ID), 및 이 네임스페이스에 대한 서버 파일 저널(148)에서 최신 리비전을 나타내는, 이 네임스페이스(예를 들어, SJ_ID)에 대한 서버 파일 저널(148)의 행을 식별할 수 있다. 커서(1974B)에서 네임스페이스와 행은 선형화된 동작(1974A)에서의 동작과 연관될 수 있다. 커서(1974B)는 특정 네임스페이스에 대한 서버 파일 저널(148)에서의 리비전의 로그에서, 선형화된 동작(1974A)이 클라이언트 디바이스(150)에 적용된 후에 및/또는 전에 서버 파일 저널(148)에서 네임스페이스의 리비전 또는 상태를 나타내는 특정 위치를 식별할 수 있다. 따라서, 커서(1974B)는 선형화된 동작(1974A) 전에 또는 후에 서버 파일 저널(148)에서 네임스페이스 및/또는 콘텐츠 아이템의 상태를 나타낼 수 있으며, 이는 선형화된 동작(1974A)이 적용되기 전에 및 후에 리비전 충돌을 피하고 리비전 순서를 추적하는 것을 도와줄 수 있다.

[0287] 도 20a는 서버 파일 저널 데이터를 선형화된 동작으로 변환하기 위한 예시적인 변환 및 선형화 프로세스를 도시한다. 이 예에서, 서버 파일 저널(148) 내의 저널(1960)은 서버 파일 저널(148)에 의해 추적된 리비전(1972)을 갖는 행(2002)을 포함한다. 저널(1960) 내의 리비전(1972)은 네임스페이스(100 및 101)(즉, NS_ID(100 및 101))와 연관된다. 일부 경우에, 서버 파일 저널(148)은 각각의 네임스페이스에 특정한 리비전을 추적하는 네임스페이스 특정 저널을 저장할 수 있다. 네임스페이스 특정 저널에서의 행(예를 들어, 2002)은 이 네임스페이스에 특정된 데이터를 포함하고, 각 행은 이 네임스페이스에 특정된 리비전을 반영한다.

[0288] 저널(1960)에서의 각 행(2002)은 이 행과 연관된 네임스페이스를 고유하게 식별하기 위한 네임스페이스 식별자 필드(NS_ID), 주어진 네임스페이스에서의 행에 매핑되고 이 네임스페이스 내에서의 동작 또는 리비전 순서를 제공하는 단조 증가하는 값을 포함하는 서버 저널 식별자 필드(SJ_ID)를 포함한다. 저널(1960)은 또한 콘텐츠 아이템의 네임스페이스-상대 경로를 식별하기 위한 경로 필드(path), 콘텐츠 아이템과 연관된 블록 또는 블록리스트를 식별하기 위한 블록 필드(Block), 및 콘텐츠 아이템의 이전 상태 또는 리비전을 나타내는 저널(1960)에서의 행(즉, SJ_ID)을 식별하기 위한 이전 리비전 필드(Prev_Rev), 및 (행이 마운트에 대응하는 경우) 마운트된 네임스페이스의 마운트 포인트에 대한 타깃 네임스페이스를 식별하기 위한 타깃 네임스페이스 필드(타깃_NS)를 포함한다. 마운트 포인트에 대응하지 않는 행(예를 들어, 리비전)에 대한 타깃_NS 필드에 대한 데이터는 없다.

[0289] 저널(1960)에서 제1 행(2002)은, 블록 "h1"에 대응하고 이전 리비전(Prev_Rev) 또는 타깃 네임스페이스(타깃_NS)를 갖지 않는, 네임스페이스 "100"(NS_ID 100)에서의 "File1"(경로 필드 값(File1))에 대한 제1 리비전(SJ_ID 1)을 식별한다. 행은 이전 리비전 또는 타깃 네임스페이스를 포함하지 않기 때문에 행으로 표현되는 리비전은 "h1" 블록과 연관된 "File1"의 네임스페이스 "100"에서의 추가에 대응한다. SJ_ID "4"를 포함하는 저널(1960)에서의 행은 네임스페이스 "100" 상의 "File1"에 대한 저널(1960)에서의 마지막 리비전을 나타내는데, 그 이유는 이 행이 네임스페이스 "100" 상의 이 "File1"에 대응하는 저널(1960)에서의 마지막 행 또는 SJ_ID이기 때문이다. SJ_ID "4"를 포함하는 이 행은 SJ_ID "1"에 추가된 후 네임스페이스 "100" 상의 "File1"이 편집되었으며 편집은 블록 "h4"에 대응하는 것을 나타낸다.

[0290] 수정(2004)은 리비전(1972)을 나타내는 수정의 일례를 도시한다. 이 예에서, 각각의 수정(2004)은 저널(1960)에서의 대응하는 행(2002)으로부터의 콘텐츠 리비전을 도시한다. 각각의 수정은 저널(1960)에서의 SJID 및 NSID, 및 저널(1960)에서의 대응하는 SJID 및 NSID와 연관된 파일에 대응한다. 이 예에서, 수정(2004)과 연관된 콘텐츠는 저널(1960)에서 블록(예를 들어, "h1", "h2", "h3", "h4")의 예시적인 콘텐츠 값을 나타낸다. 수정(2004)에서의 콘텐츠 값은 각 리비전과 연관된 콘텐츠에 대한 예시적인 수정을 도시하기 위해 예시를 위한 목적으로 제공된다.

[0291] 예를 들어, 수정(2004)에서의 제1 수정은 저널(1960)에서 SJID "1" 및 NSID "100"을 나타내고, 네임스페이스 "100"에서 "File1"이 추가되는 것을 도시한다. 콘텐츠 "aaa"는 NSID "100"의 SJID "1"에서 "File1"에 대한 "h1" 값을 나타낸다. 수정(2004)은 또한 "aa2"(예를 들어, "h4")로 수정된 네임스페이스 "100"의 "File1"과 연관된 콘텐츠 "aaa"(예를 들어, "h1")을 보여주는 저널(1960)에서의 SJID "4" 및 NSID "100"을 나타내는 네임스페이스 "100"에서의 "File1"의 편집을 보여준다.

[0292] 변환(1970)에서, 저널(1960)에서의 행(2002)의 리비전(1972)은 선형화된 동작(1974A)으로 변환된다. 선형화된

동작(1974A)은 저널(1960) 내의 리비전(1972)으로부터 생성되고, 선형화 후의 수정(2004)을 나타낸다. 선형화된 동작(1974A)에 의해 도시된 바와 같이, 선형화된 동작(1974A)에서의 동작은 다수의 리비전(1972) 및/또는 수정(2004), 또는 단일 리비전(1972) 및/또는 수정(2004)에 기초할 수 있다.

[0293] 예를 들어, 수정(2004)은 저널(1960)에서의 SJID "1" 및 NSID "100"에 대응하는 네임스페이스 "100"에 "File1"을 추가하는 리비전, 및 저널(1960)에서 SJID "4" 및 NSID "100"에 대응하는 네임스페이스 "100"에서의 "File1"을 편집하는 리비전을 도시한다. 추가 리비전은 "File1" 및 NSID "100"과 연관된 콘텐츠 값 "aaa"(예를 들어, "h1") 및 "File1" 및 NSID "100"에 대한 이전 리비전의 부재로부터 추론될 수 있다. 즉, 콘텐츠 "aaa"는 콘텐츠(예를 들어, "h1")가 추가 또는 편집되었다는 것을 나타내며, "File1" 및 NSID "100"에 대한 이전 리비전의 부재는 콘텐츠 "aaa"가 콘텐츠(예를 들어, "h1")가 편집되고 있는 것이 아니라 추가되고 있는 것임을 암시한다. 편집 리비전은 "File1" 및 NSID "100"과 연관된 콘텐츠 값 "aa2"(예를 들어, "h4"), 및 "File1" 및 NSID "100"과 연관된 이전 리비전(SJID "1" 및 NSID "100")으로부터 추론될 수 있다. 다시 말해, "File1" 및 NSID "100"과 연관된 콘텐츠 "aaa"로부터 "aa2"로의 변경은 콘텐츠 "aa2"가 편집을 나타낸다는 것임을 암시한다.

[0294] 선형화된 동작(1974A)에서, NSID "100"에 대한 SJID "1" 및 SJID "4"에 대응하는 추가 및 편집 수정(2004)은 "File1"과 연관된 콘텐츠 값을 "aaa"(예를 들어, "h1")로부터 "aa2"(예를 들어, "h4")로 편집하는 단일 선형화된 동작(편집 동작)으로 변환될 수 있다. "File1"의 콘텐츠(예를 들어, "h1")를 "aa2"(예를 들어, "h4")로 편집하는 단일 선형화된 동작은 콘텐츠 "aaa"(예를 들어, "h1")와 연관된 "File1"을 네임스페이스 "100"에 추가하는 수정뿐만 아니라, 네임스페이스 "100"에서 "File1"과 연관된 콘텐츠 "aaa"(예를 들어, "h1")를 "aa2"(예를 들어, "h4")로 편집하는 수정을 반영한다. 따라서, 이 선형화된 동작은 2개의 수정(2004) 및 수정(1972)에서의 2개의 대응하는 리비전에 기초한다.

[0295] 저널(1960)에서 SJID "2" 및 NSID "100"에 대응하는 수정(2004)에서의 수정은 콘텐츠 "bbb"(예를 들어, "h2")와 연관된 "File2"를 네임스페이스 "100"에 추가하는 리비전을 나타낸다. 이 수정은 네임스페이스 "100" 상에서 "File2"에 대응하는 저널(1960)로부터의 유일한 리비전(1972)을 나타낸다. 따라서, 선형화된 동작(1974A)은 네임스페이스 "100" 상의 "File2"에 대한 단일 동작을 포함하는데, 이 단일 동작은 콘텐츠 "bbb"(예를 들어, "h2")와 연관된 "File2"를 네임스페이스 "100"에 추가하고, 단일 수정(2004)(네임스페이스 "100" 상에서의 "File2"의 추가) 및 리비전(1972)에 기초한다.

[0296] 이 예에서의 수정(2004)은 또한 저널(1960) 내의 SJID "3" 및 NSID "100"에 대응하는 콘텐츠 "ccc"(예를 들어, "h3")와 연관된 "File3"을 네임스페이스 "100"에 추가하는 수정, 및 저널(1960) 내의 SJID "5" 및 NSID "100"에 대응하는 "File3"을 네임스페이스 "100"으로부터 삭제("-1"로 표현됨)하는 것을 포함한다. 따라서 리비전(1972)은 네임스페이스 "100" 상의 "File3"과 연관된 2개의 수정(2004)을 포함한다. "File3" 및 네임스페이스 "100"과 연관된 저널(1960)에서의 마지막 리비전은 저널(1960) 내의 SJID "5" 및 NSID "100"을 나타내는 삭제 수정에 대응하기 때문에, 리비전(1972)으로부터의 "File3" 및 네임스페이스 "100"과 연관된 추가 및 삭제 수정(2004)은 네임스페이스 "100"로부터 "File3"을 삭제하는 단일 동작으로 선형화될 수 있다. 따라서, 선형화된 동작(1974A)은 "File3" 및 네임스페이스 "100"에 대한 단일 동작을 포함하는데, 단일 동작은 네임스페이스 "100"으로부터 "File3"을 삭제하는 단일 동작이다.

[0297] 저널(1960)에서의 NSID "100"에 대한 SJID "6" 및 "7" 및 NSID "101"에 대한 SJID "1"은 "Dir"이 네임스페이스 "100"에 추가되고 나중에 네임스페이스 "100"로부터 네임스페이스 "101"로 이동되는 것을 나타낸다. 예를 들어, SJID "6" 및 NSID "100"은 "Dir" 및 네임스페이스 "100"을 식별하며, "Dir"이 SJID "6"에서 네임스페이스 "100"에 추가되었다는 것을 나타내는 이전 리비전을 포함하지는 않는다. SJID "7"은 "Dir"이 블록 필드("-"), 이전 리비전 필드(SJID "6") 및 타깃 네임스페이스 필드("101")에 의해 반영되는 바와 같이 네임스페이스 "100"로부터 네임스페이스 "101"로 이동되었다는 것을 식별한다. NSID "101"에 대한 SJID "1"은 "Dir" 및 네임스페이스 "101"에 대한 이전 행 또는 리비전의 부재에 의해 나타낸 바와 같이 "Dir"이 네임스페이스 "101"에 추가되는 것을 식별한다. NSID "100"에서의 SJID "6" 및 "7" 및 NSID "8"에서의 SJID "1"에서의 추가 및 이동 리비전은 3개의 수정(2004), 즉 SJID "6" 및 NSID "100"에 대응하는 네임스페이스 "100"에 "Dir"의 추가; SJID "7" 및 NSID "100"에 대응하는 네임스페이스 "100"으로부터 "Dir"의 삭제; 및 SJID "1" 및 NSID "101"에 대응하는 네임스페이스 "101"에 "Dir"의 추가로 도시된다.

[0298] 저널(1960)에서의 NSID "100"의 SJID "6" 및 "7"에 각각 대응하는, "Dir" 및 네임스페이스 "100"의 추가 및 삭제 수정(2004)은 네임스페이스 "100"으로부터 "Dir"을 삭제하는 단일 동작으로 선형화되는데, 그 이유는 "Dir" 및 네임스페이스 "100"에 대응하는 저널(1960)에서의 마지막 리비전이 SJID "7" 및 NSID "100"에서 네임스페이

스 "100"으로부터 "Dir"을 삭제하는 것이기 때문이다. 저널(1960)에서의 SJID "1" 및 NSID "101"에 대응하는, 네임스페이스 "101"에 "Dir"의 추가는 "Dir" 및 네임스페이스 "101"에 대응하는, 유일한 수정(2004) 및 리비전(1972)이다. 따라서, 추가는 "Dir" 및 네임스페이스 "101"에 대한 단일 마운트 동작으로서 선형화된 동작(1974A)에서 제공된다. 따라서 NSID "100"에서의 SJID "6" 및 "7" 및 NSID "101"에서의 SJID "1"에 대응하는 수정(1972)으로부터 3개의 수정(2004)(즉, 네임스페이스 "100" 상의 "Dir"의 추가 및 삭제, 및 네임스페이스 "100" 상의 "Dir"의 추가)은 선형화된 동작(1974A)에서 2개의 동작, 즉 네임스페이스 "101"에서의 "Dir"에 대한 삭제 동작과 네임스페이스 "101"에서의 "Dir"에 대한 마운트 동작으로 선형화된다.

[0299] 위에서 예시된 바와 같이, 선형화된 동작(1974A)은 "File1" 및 네임스페이스 "100"에 대한 편집 동작, "File2" 및 네임스페이스 "100"에 대한 추가 동작, 네임스페이스 "100"에서의 "File3"의 삭제 동작, 네임스페이스 "100"에서의 "Dir"에 대한 삭제 동작, 및 네임스페이스 "101"에 "Dir"을 추가하기 위한 마운트 동작을 포함한다. 선형화된 동작(1974A)에서의 이들 동작은 리비전(1972)으로부터 생성되고, 저널(1960)에서의 각 콘텐츠 아이템의 최신 상태를 반영한다. 파일 저널 인터페이스(1902)는 선형화된 동작(1974A)을 생성하고, 선형화된 동작(1974A)을 클라이언트 디바이스(150)에 송신하여, 클라이언트 디바이스(150)가 저널(1960)에서의 리비전(1972)으로부터 최신 상태를 포함하는 것을 보장할 수 있다.

[0300] 클라이언트 디바이스(150)에 선형화된 동작(1974A)을 제공할 때, 파일 저널 인터페이스(1902)는 클라이언트 디바이스(150)에 대한 선형화된 동작(1974A)과 함께 커서(1974B)를 포함할 수 있다. 커서(1974B)는 저널(1960)에서의 각 네임스페이스(NSID)에 대한 최종 리비전(SJID)을 식별할 수 있다. 일부 실시형태에서, 커서(1974B)는 또한 사용자 ID를 포함하는 FSAuth 토큰, 및 커서에 제공된 NS_ID에 대한 마지막으로 관찰된 액세스 권한을 포함할 수 있다. 각 네임스페이스에 대한 최종 리비전은 각 네임스페이스에 대해 클라이언트 디바이스(150)에 전송된 최신 리비전에 대응하는 저널(1960)에서의 위치를 나타낼 수 있다.

[0301] 일부 경우에, 커서(1974B)는 선형화된 동작(1974A)에서의 각각의 동작을 저널(1960)에서의 네임스페이스(NSID)와 행(SJID)에 매핑할 수도 있다. 동작과 연관된 네임스페이스 및 행은 이 동작에 대응하는 저널(1960)에서의 위치를 나타낼 수 있다. 즉, 동작과 연관된 네임스페이스 및 행은 이 동작으로 표현되는 저널(1960)에서의 리비전 번호를 나타낼 수 있다. 커서(1974B)에서의 네임스페이스 및 행은 선형화된 동작(1974A)과 연관된 각 네임스페이스 및 콘텐츠 아이템에 대한 저널(1960)에서의 최신 상태에 대응한다. 커서(1974B)는, (예를 들어 동작 데이터(1932)를 통해) 클라이언트 디바이스(150)로부터 하나 이상의 네임스페이스 및/또는 콘텐츠 아이템으로 변경을 적용하려고 시도할 때, 하나 이상의 네임스페이스 및/또는 콘텐츠 아이템에 대해 클라이언트 디바이스(150)에 의해 획득된 최신 상태 또는 리비전을 파일 저널 인터페이스(1902)에게 식별시키기 위한 클라이언트 디바이스(150)에 대한 도구로서 클라이언트 디바이스(150)로 제공될 수 있다. 파일 저널 인터페이스(1902)가 클라이언트 디바이스(150)로부터 커서(1974B)를 수신할 때, 파일 저널 인터페이스는 커서(1974B)를 사용하여 저널(1960)에서 클라이언트 디바이스(150)의 위치(예를 들어, 클라이언트 디바이스(150)에 의해 획득된 저널(1960)로부터 최신 리비전)를 식별하고 클라이언트 디바이스(150)로부터의 동작에 의해 야기된 충돌을 검출 또는 회피할 수 있다.

[0302] 예를 들어, 파일 저널 인터페이스(1902)가 네임스페이스 "100"에서의 "File1"을 수정하는 동작을 클라이언트 디바이스(150)로부터 수신하면, 파일 저널 인터페이스(1902)는 동작과 함께 클라이언트 디바이스(150)로부터 수신하는 커서(1974B)를 사용하여 저널(1960)이 클라이언트 디바이스(150)로부터의 커서(1974B)에서 식별된 리비전보다 네임스페이스 "100"에서의 "File1"에 대해 임의의 더 새로운 리비전을 갖는 것이 있는지 여부를 검사할 수 있다. 커서(1974B)에서의 리비전이 저널(1960)에서의 가장 현재의 리비전인 경우, 파일 저널 인터페이스(1902)는 네임스페이스 "100"에서의 "File1"에 대해 저널(1960)에서의 새로운 리비전(예를 들어, NSID "100"에서의 SJID "8")으로서 편집 동작을 커밋할 수 있다.

[0303] 대안적으로, 커서(1974B)에서의 리비전이 네임스페이스 "100"에서의 "File1"에 대한 저널(1960)에서의 가장 현재의 리비전이 아닌 경우, 파일 저널 인터페이스(1902)는 클라이언트 디바이스(150)로부터의 편집 동작이 네임스페이스 "100"에서의 "File1"에 대한 저널(1960)에서의 가장 현재의 리비전에 기초하지 않은 것이라고 결정할 수 있다. 예를 들어, 커서(1974B)가 저널(1960) 내의 SJID "4" 및 NSID "100"을 식별하고, 파일 저널 인터페이스(1902)가 저널(1960)이 네임스페이스 "100"에서의 "File1"에 대한 SJID "12" 및 NSID "100"에서의 리비전을 포함하는 것이라고 결정하는 경우 파일 저널 인터페이스(1902)는 클라이언트 디바이스(150)로부터의 편집 동작이 네임스페이스 "100" 상의 "File1"의 더 오래된 버전(예를 들어, SJID "4" 및 NSID "100")에 속하는 것이라고 결정할 수 있고, 편집 동작은 수정된 이후 갖는 파일을 편집할 때 충돌을 야기할 수 있다. 파일 저널 인터페이스(1902)는 편집 동작에 의해 생성된 이러한 충돌을 검출하고, 편집 동작을 거부하거나, 충돌을 조정하려고

시도하거나, 또는 클라이언트 디바이스(150)에 최신 리비전을 제공하고, 클라이언트 디바이스(150)가 충돌을 조정하는 것을 허용할 수 있다.

[0304] 파일 저널 인터페이스(1902)가 선형화된 동작을 클라이언트 디바이스(150)에 전송할 때마다, 파일 저널 인터페이스는 각각의 네임스페이스 및/또는 콘텐츠 아이템에 대한 저널(1960)에서의 각각의 위치를 식별하는, 본 명세서에 설명된 바와 같은 커서를 포함할 수 있다. 유사하게, 클라이언트 디바이스(150)가 파일 저널 인터페이스(1902)에 동작을 전송하는 임의의 시간에, 파일 저널 인터페이스는 파일 저널 인터페이스(1902)가 클라이언트 디바이스(150)에서의 상태를 저널(1960)에서의 상태와 매핑하는데 사용할 수 있는 최신의 커서를 포함할 수 있다.

[0305] 이 예에서 저널(1960)은 다수의 네임스페이스를 갖는 저널을 도시한다. 전술한 바와 같이, 일부 예에서, 서버 파일 저널(148)은 네임스페이스 특정 저널을 유지할 수 있다. 커서(1974B)는 각 네임스페이스에 대한 최신 리비전을 나타내기 위해 각 네임스페이스에 대한 SJID 및 NSID를 포함할 수 있다. 커서(1974B)에 기초하여, 파일 저널 인터페이스(200)는 본 명세서에서 추가로 설명되는 바와 같이, 다수의 저널이 유지되는 실시형태에서, 다수의 저널을 조회하고/하거나 다수의 저널로부터 리비전을 검색할 수 있다.

[0306] 도 20b는 클라이언트 디바이스(150)로부터의 동작 데이터(1932)를 서버 파일 저널(148)에서의 저널(1960)에 대한 리비전(1972)으로 변환하기 위한 선형화(2010)를 위한 예시적인 프로세스를 도시한다. 클라이언트 디바이스(150)는 동작 데이터(1932)를 파일 저널 인터페이스(1902)에 제공할 수 있다. 이 예에서 동작 데이터(1932)는 콘텐츠 아이템 편집, 추가, 이름 변경, 이동, 마운트, 또는 삭제 동작과 같은, 클라이언트 디바이스(150)에서의 동작(2012)을 포함한다. 경우에 따라 동작(2012)은 동일한 콘텐츠 아이템에 대한 다수의 동작을 포함할 수 있다. 예를 들어, 동작(2012)은 네임스페이스 "100" 상의 "File4"를 편집하는 동작 및 네임스페이스 "100"으로부터 "File4"를 삭제하는 동작을 포함할 수 있다.

[0307] 동작 데이터(1932)는 또한 파일 저널 인터페이스(1902)로부터 클라이언트 디바이스(150)에 의해 이전에 수신된 커서(1974B)를 포함한다. 커서(1974B)는 하나 이상의 네임스페이스 및/또는 콘텐츠 아이템에 대한 저널(1960)에서의 상태(예를 들어, NSID 및 SJID) 또는 최신 리비전을 식별할 수 있다. 클라이언트 디바이스(150)는 동작(2012)에 대한 기준 포인트로서 파일 저널 인터페이스(1902)에 커서(1974B)를 제공할 수 있다. 이 예에서, 커서(1974B)는 SJID "9"로 표시된 네임스페이스 "100"에 대한 최신 상태를 제공한다.

[0308] 일부 경우에, 커서는 콘텐츠 관리 시스템(110)에 의해 암호로 서명되며, 이는 파일 저널 인터페이스(1902)가 커서가 무단 변경되지 않았다고 결정할 수 있게 한다. 또한, 클라이언트 디바이스(150)는 네임스페이스에 대한 서버 파일 저널(148)로부터 가장 최근의 리비전을 수신한 경우 서버 파일 저널(148)에 대한 리비전을 커밋하기 때문에, 파일 저널 인터페이스(1902)는 NS_ID에 대한 마지막으로 관찰된 액세스 권한이 여전히 유효하다는 것을 수락할 수 있고, 따라서 클라이언트 디바이스(150)는 네임스페이스에 액세스할 수 있다.

[0309] 파일 저널 인터페이스(1902)는 동작(2012) 및 커서(1974B)를 수신하고 선형화(2010)를 수행하여, 클라이언트 디바이스(150)로부터의 동작(2012)을 저널(1960)에 대한 리비전(1972)으로 선형화 및 변환할 수 있다. 동작(2012)에 기초하여, 파일 저널 인터페이스(1902)는 동작 로그(2014)를 생성할 수 있다. 로그(2014)는 저널(1960) 내의 각 네임스페이스(들)에 매핑된 동작(2012)으로부터의 동작 리스트를 포함할 수 있다. 경우에 따라 로그(2014)는 앞에서 설명한 바와 같이 동작(2012)으로부터 생성된 선형화된 동작을 포함할 수 있다.

[0310] 파일 저널 인터페이스(1902)는 커서(1974B)를 사용하여, 로그(2014)에서의 동작을 반영하기 위해 저널(1960)을 업데이트하기 전에, 동작(2012)이 저널(1960)의 최신 상태 또는 리비전을 반영하는지 검증할 수 있다. 파일 저널 인터페이스(1902)가 커서(1974B)가 로그(2014)와 연관된 네임스페이스 및/또는 콘텐츠 아이템에 대해 저널(1960)에서의 최신 상태 또는 리비전을 반영한다고 확인하면, 파일 저널 인터페이스(1902)는 로그(2014)에 기초하여 리비전(1972)을 저널(1960)에 추가할 수 있다. 리비전(1972)은 로그(2014)에서의 동작과 연관된 각 콘텐츠 아이템 및/또는 네임스페이스의 최신 상태 또는 리비전을 포함할 수 있다.

[0311] 로그(2014)의 동작은 "File5"에 대한 추가 및 편집 동작을 포함한다. 따라서, 리비전(1972)은 "File5"의 편집을 포함하고, 이 파일 저널 인터페이스(1902)는 이것을 "File5"의 최신 상태(즉, 추가 및 편집 동작이 선형화된 방식으로 "File5"에 적용된 후의 상태)로서 저널(1960)에 기입할 수 있다. 로그(2014)에서의 동작은 또한 "Dir2"에 대한 추가 동작뿐만 아니라 네임스페이스 "100" 상의 "File4"에 대한 편집 및 삭제 동작을 포함한다. 따라서 리비전(1972)은 "Dir2" 및 "File4"의 최신 상태로서, 각각 네임스페이스 "100"에 "Dir2"를 추가하는 동작 및 네임스페이스 "100"으로부터 "File4"를 삭제하는 동작을 포함할 수 있다.

[0312] 도 20b에서, 저널(1960)에 도시된 리비전(1972)은 동작(2012)과 연관된 각 콘텐츠 아이템("File4", "File5", "Dir2")의 최신 상태를 반영한다. 그러나, 일부 경우에 파일 저널 인터페이스(1902)는 로그(2014)로부터 유래되는 각 네임스페이스 및/또는 콘텐츠 아이템의 최신 상태 리비전뿐만 아니라 또한 최신 상태 또는 리비전으로 이어지는 임의의 이전 상태 또는 리비전을 반영하기 위해 로그(2014)로 표시되는 모든 리비전을 저널(1960)에 기입할 수 있다는 것에 주목해야 한다. 예를 들어, 파일 저널 인터페이스(1902)는 동작(2012)으로부터 "File4"의 리비전의 전체 순서를 저널(1960)에 나타내기 위해 동작(2012)으로부터 최신 상태를 반영하는 "File4"의 편집만을 기입하는 것과 달리, "File4"의 편집을 위해 저널(1960)에서의 리비전 및 "File4"의 삭제를 위한 후속 리비전을 기입할 수 있다.

[0313] 파일 저널 인터페이스(1902)는 로그(2014)에서의 동작을 리비전(1972)으로 변환하고, 리비전(1972)을 포함하도록 저널(1960)을 업데이트할 수 있다. 파일 저널 인터페이스(1902)는 저널(1960) 내의 각 행에서 리비전(1972)을 저널(1960)에 기입할 수 있다. 파일 저널 인터페이스(1902)는 저널(1960)에서 그 다음 이용 가능한 행(예를 들어, SJID)에 리비전(1972)을 추가할 수 있다. 일부 경우에, 파일 저널 인터페이스(1902)는 선형화(2010) 및/또는 각각의 타임스탬프 또는 클록에 기초하여 결정될 수 있는 상대 순서에 기초하여 리비전(1972)을 추가할 수 있다.

[0314] 도 20b에 도시된 바와 같이, 네임스페이스 "100"에서의 "File4"의 삭제 동작은 네임스페이스 "100"에 대한 행 "11" 또는 SJID "11"에 포함된다. 저널(1960)의 SJID "11"에서의 리비전은 블록 필드에서의 마이너스 기호로 반영된 바와 같이 네임스페이스 "100"에서의 "File4"가 삭제되었다는 것을 나타내고, SJID "9"를 네임스페이스 "100"에서의 "File4"에 대해 저널(1960)에서의 이전 리비전으로 식별한다. "Dir2"의 추가 및 "File5"의 편집은 각각 행 또는 SJID 12 및 14에 포함된다.

[0315] 도 20b에서의 저널(1960)은 로그(2014)에서 수정된 각 콘텐츠 아이템의 상태를 반영하기 위해 로그(2014) 및 커서(1974B)에 기초하여 리비전(1972)을 포함하도록 업데이트되었다. 저널(1960) 내의 각 행에서의 경로 필드는 연관된 네임스페이스(예를 들어, 네임스페이스 "100") 내의 콘텐츠 아이템을 식별한다. 행의 경로 필드는 로그(2014)에서의 대응하는 동작으로부터의 파일 및 네임스페이스에 기초한다. 저널(1960)에서의 블록 필드는 콘텐츠 아이템을 나타낸다. 일부 경우에, 블록 필드는 각각의 콘텐츠 아이템 또는 데이터 블록의 해시를 포함할 수 있다. 블록 필드는 콘텐츠 아이템이 삭제되었고/되었거나 디렉토리, 폴더, 마운트 등인 경우 비어 있을 수 있다.

[0316] 로그(2014) 및 커서(1974B)에 기초하여 리비전(1972)을 포함하도록 저널(1960)을 업데이트하는 경우, 변환 서비스(1904)는 저널(1960)의 경로 필드에 포함할 각각의 콘텐츠 아이템의 경로를 식별할 수 있다. 일부 경우에, 변환 서비스(1904)는 콘텐츠 아이템의 식별자(예를 들어, File ID)를 콘텐츠 아이템의 경로(예를 들어, /directory/filename)로 변환할 수 있다. 예를 들어, 클라이언트 디바이스(150)는 콘텐츠 아이템에 대한 각각의 경로를 추적하거나 계산할 필요없이 식별자를 사용하여 콘텐츠 아이템(예를 들어, 동작 데이터(1932)에서의 콘텐츠 아이템)을 식별할 수 있다. 저널(1960)은 대신 콘텐츠 아이템의 경로를 사용하여 콘텐츠 아이템을 식별할 수 있다. 변환 서비스(1904)는 클라이언트 디바이스(150)로부터의 콘텐츠 아이템의 식별자를 사용하여 저널(1960)에 대한 콘텐츠 아이템의 경로를 계산하고, 콘텐츠 아이템에 대해 계산된 경로를 사용하여 저널(1960)을 업데이트할 수 있다. 변환 서비스(1904)는 또한 콘텐츠 아이템의 경로에 기초하여 콘텐츠 아이템의 식별자를 얻기 위해 역 변환을 수행할 수 있고, 클라이언트 디바이스(150)와 통신하는 콘텐츠 아이템을 참조할 때 콘텐츠 아이템의 식별자를 사용할 수 있다.

[0317] 예를 들어, 변환 서비스(1904)는 저널(1960)에서의 경로, 저널(1960)에서의 NSID, 및/또는 저널(1960)에서의 디렉토리 필드(또는 서버 파일 저널(148)에서의 다른 곳)를 사용하여 콘텐츠 아이템을 식별하고 이 콘텐츠 아이템의 식별자(예를 들어, File ID)를 획득할 수 있다. 파일 저널 인터페이스(1902)가 이 콘텐츠 아이템에 관한 업데이트 또는 정보를 클라이언트 디바이스(150)에 전송하면, 파일 저널 인터페이스(1902)는 콘텐츠 아이템의 식별자를 클라이언트 디바이스(150)에 제공할 수 있고, 클라이언트 디바이스(150)는 이 식별자를 사용하여 콘텐츠 아이템의 경로와 함께 또는 경로 없이 콘텐츠 아이템을 식별할 수 있다.

[0318] 전술한 바와 같이, 동작(2012)으로부터 리비전(1972)을 저널(1960)에 기입하기 전에, 파일 저널 인터페이스(1902)는 커서(1974B)가 동작(2012)과 연관된 각각의 네임스페이스 및/또는 콘텐츠 아이템에 대한 저널(1960)에서의 최신 상태 또는 리비전을 반영하는지 여부를 검사할 수 있다. 일부 경우에 커서(1974B)가 저널(1960)에서의 최신 상태 또는 리비전을 반영한다는 것을 확인한 후, 파일 저널 인터페이스(1902)는 동작(2012)으로부터 생성된 리비전이 저널(1960)에서의 현존하는 리비전과 충돌하지 않는다는 것을 보장하기 위해 제2 검사를 수행할

수도 있다. 예를 들어, 저널(1960)에서의 네임스페이스 "100"에서의 SJID "5"가 "File5"의 삭제 동작을 나타내는 경우, 클라이언트 디바이스로부터 파일 저널 인터페이스(1902)에 의해 수신된 동작(2012)으로부터 방출된 SJID "14"에 도시된 "File5"의 편집 리비전(1972)은 "File5"가 SJID "5"에서 삭제된 경우에도 "File5"를 편집하려고 시도하는 것에 의해 충돌을 일으킬 수 있다. 따라서, 파일 저널 인터페이스(1902)는 이 예에서 편집 동작 및 리비전을 거부하고, 편집 동작이 유효하지 않다는 것을 클라이언트 디바이스(150)에 전달할 수 있다. 파일 저널 인터페이스(1902)는 커서(1974B)를 업데이트하고 업데이트된 커서를 클라이언트 디바이스(150)에 제공하여, 필요에 따라 "File5"(및 임의의 다른 콘텐츠 아이템)에 대한 저널(1960)에서의 최신 상태 또는 리비전을 클라이언트 디바이스(150)에 통지할 수 있다.

[0319] 새로운 리비전이 저널(1960) 및/또는 서버 파일 저널(148)에 추가될 때, 파일 저널 인터페이스(1902)는 새로운 리비전을 보고하고 새로운 리비전을 클라이언트 디바이스(150)와 동기화하기 위해 업데이트된 커서를 클라이언트 디바이스(150)에 전송할 수 있다. 클라이언트 디바이스(150)는 클라이언트 디바이스(150)에서 커서에 업데이트를 요청할 수도 있다. 클라이언트 디바이스(150)는 파일 저널 인터페이스(1902)로부터 수신된 마지막 커서의 사본을 클라이언트 디바이스(150) 상의 콘텐츠 아이템의 상태 및/또는 클라이언트 디바이스(150)에 의해 획득된 최종 리비전(들)을 나타내는 저널(1960)에서의 클라이언트 디바이스(150)의 위치를 반영하는 것으로 저장할 수 있다.

[0320] 도 20c는 서버 파일 저널(148)로부터 클라이언트 디바이스(150)에 대한 동작으로 리비전을 변환하기 위한 예시적인 방법을 도시한다. 단계(2050)에서, 파일 저널 인터페이스(1902)는 콘텐츠 관리 시스템(110)에 등록된 사용자 계정에 대해 클라이언트 디바이스(150)에 저장된 하나 이상의 콘텐츠 아이템과 연관된 복수의 리비전(예를 들어, 1972)을 서버 파일 저널(148)에서 리비전의 저널(1960)로부터 검색한다. 각 리비전은 네임스페이스, 폴더, 파일 또는 임의의 콘텐츠 아이템을 수정할 수 있다. 또한 각 리비전은 네임스페이스 및 이 네임스페이스의 저널 ID(SJID)와 연관될 수 있다.

[0321] 일부 경우에, 파일 저널 인터페이스(1902)는 저널(1960)이 클라이언트 디바이스(150)에서 이용할 수 없는 리비전을 포함하도록 업데이트되었다는 결정에 기초하여 저널(1960)로부터 복수의 리비전을 검색할 수 있다. 예를 들어, 파일 저널 인터페이스(1902)는 저널(1960)에 추가된 새로운 리비전을 추적하고 및/또는 클라이언트 디바이스(150)에서 커서와 저널(1960)에서의 리비전을 비교할 수 있다. 일부 경우에, 파일 저널 인터페이스(1902)는 저널(1960)을 조회하여 복수의 리비전을 검색하고 및/또는 저널(1960)에서 이용 가능한 리비전을 검사할 수 있다.

[0322] 단계(2052)에서, 파일 저널 인터페이스(1902)는 복수의 리비전과 연관된 각각의 콘텐츠 아이템의 각각의 리비전 세트에 기초하여 각각의 동작을 결정한다. 예를 들어, 파일 저널 인터페이스(1902)는 콘텐츠 아이템의 임의의 리비전을 선형화하고 이 콘텐츠 아이템에 대한 하나 이상의 각각의 동작으로 리비전을 변환할 수 있다. 일부 경우에, 파일 저널 인터페이스(1902)는 또한 다수의 동작이 선형 방식으로 실행될 때 이 콘텐츠 아이템에 대한 다수의 동작을 콘텐츠 아이템의 상태 또는 수정을 정의 또는 반영하는 단일 동작으로 변환할 수 있다.

[0323] 일부 경우에, 복수의 리비전에 대한 각각의 동작을 계산할 때, 파일 저널 인터페이스(1902)는 특정 콘텐츠 아이템과 연관된 리비전의 수 및/또는 이러한 리비전과 연관된 콘텐츠 아이템의 유형에 기초하여 추론 또는 계산을 수행할 수 있다. 예를 들어, 복수의 리비전이 콘텐츠 아이템에 대한 단일 리비전을 포함하는 경우, 파일 저널 인터페이스(1902)는 단일 리비전(예를 들어, 리비전(1972)) 및/또는 리비전과 연관된 블록 또는 콘텐츠(예를 들어, 저널(1960)의 행(2002)에서 블록 또는 콘텐츠)로부터 이 리비전에 의해 표현된 콘텐츠 아이템의 수정 유형(예를 들어, 2004)을 추론할 수 있고, 이 리비전에 의해 표현된 수정 유형에 기초하여 이 콘텐츠 아이템에 대한 각각의 동작을 계산할 수 있다.

[0324] 설명하기 위해, 도 4a에 도시된 바와 같이, 수정(2004)은 SJID "1" 및 NSID "100"에 대응하는 네임스페이스 "101"에서 "Dir"에 대한 수정을 도시한다. 이 수정은 네임스페이스 "101"에 대한 유일한 수정(2004) 및 수정(1972)이다. 따라서, 파일 저널 인터페이스(1902)는 네임스페이스 "101"에서 "Dir"을 나타내는 수정이 "Dir"을 포함하도록 수정되거나 리비전되는 네임스페이스 "101"의 제1 인스턴스를 나타내기 때문에 "Dir"의 추가 또는 마운트인 것으로 추론할 수 있다. "Dir"은 저널(1960)에서의 블록 필드로 도시된 바와 같이 디렉토리 또는 폴더이기 때문에 수정은 디렉토리 또는 폴더의 추가 또는 마운트일 수 있다. "Dir"이 네임스페이스인 경우 수정은 네임스페이스 "101"에서 네임스페이스 "Dir"의 마운트를 나타낼 수 있다. 다른 한편으로, "Dir"이 저널(1960)에서의 블록 필드에 기초하여 결정될 수 있는 특정 콘텐츠 또는 블록과 연관된 파일인 경우, "Dir"에 대한 수정은 네임스페이스 "101"에 대한 "Dir" 파일의 추가일 수 있다. 예를 들어 SJID "1"과 NSID "101"이 대신 "h1"과 연

관된 "File1"을 나타내는 경우 대응하는 수정은 네임스페이스 "101"에 대한 "File1"의 추가일 수 있다.

[0325] 따라서, 저널(1960)에서의 리비전(1972)과 연관된 콘텐츠 또는 블록 필드가 삭제(예를 들어, 블록 또는 콘텐츠 필드에서의 마이너스 기호)를 나타내지 않는 한, 콘텐츠 아이템의 제1 또는 유일한 리비전에 대한 각각의 동작은 콘텐츠 아이템이 네임스페이스인지 또는 다른 유형의 콘텐츠 아이템인지 여부에 따라 마운트 또는 추가 동작을 나타낼 수 있다. 이는 편집, 마운트 해제 또는 삭제 동작과 같은 다른 동작이 연관된 콘텐츠 아이템을 마운트하거나 추가하기 위한 이전 리비전을 포함하도록 예상된다는 가정에 기초한다. 콘텐츠 아이템이 이와 연관된 이전 리비전을 가지고 있지 않은 경우, 파일 저널 인터페이스(1902)는 콘텐츠 아이템과 연관된 리비전이 편집, 마운트 해제 또는 삭제 동작이 아니라 오히려 추가 또는 마운트 동작이라고 추론할 수 있다.

[0326] 일부 경우에, 파일 저널 인터페이스(1902)는 이 콘텐츠 아이템 및 연관된 네임스페이스에 대한 다수의 리비전(1972)에 기초하여 콘텐츠 아이템에 대한 동작을 계산할 수 있다. 예를 들어, 파일 저널 인터페이스(1902)는 콘텐츠 아이템의 추가 또는 마운트를 나타내는 리비전, 및 삭제, 편집 또는 마운트 해제를 나타내는 후속 리비전으로부터 삭제, 편집 또는 마운트 해제 동작을 추론할 수 있다. 설명하기 위해, 도 20a에 도시된 바와 같이, 파일 저널 인터페이스(1902)는 저널(1960)에서 네임스페이스 "100"에 대한 SJID "1" 및 "4"에 대응하는 리비전(1972) 및 다수의 수정(2004)에 기초하여 네임스페이스 "100"에서의 "File1"에 대한 편집 동작을 계산한다. SJID "1" 및 "4"는 수정(2004)에서 콘텐츠 값 "aaa" 및 "aa2"를 나타내는 블록 "h1" 및 "h4"를 포함하므로, 파일 저널 인터페이스(1902)는 SJID "1"이 추가 동작을 나타내고 SJID"4"는 편집 동작을 나타내어 결과 상태는 SJID "4"에서 편집 동작에 기초한다고 결정할 수 있다.

[0327] 각각의 동작에 기초하여, 단계(2054)에서, 파일 저널 인터페이스(1902)는 각각의 콘텐츠 아이템에 대해 선형화된 동작(예를 들어, 1972) 세트를 생성한다. 선형화된 동작 세트는 저널(1960) 내의 복수의 리비전에 기초하여 각 콘텐츠 아이템의 수정(2004)을 반영할 수 있다. 파일 저널 인터페이스(1902)는 상대적인 클록 및/또는 인과관계에 기초하여 각 콘텐츠 아이템에 대해 계산된 각각의 동작을 선형화함으로써 복수의 리비전(1972)을 선형화된 동작 세트(324A)로 변환할 수 있다.

[0328] 단계(2056)에서, 파일 저널 인터페이스(1902)는 선형화된 동작 세트로 표현된 저널(1960)에서의 위치를 식별하는 커서(예를 들어, 324B)를 생성한다. 단계(2058)에서, 파일 저널 인터페이스(1902)는 선형화된 동작 세트 및 커서를 클라이언트 디바이스(150)에 전송한다. 커서는 각각의 네임스페이스 및/또는 동작에 대한 각각의 네임스페이스 식별자(NSID) 및 저널 식별자(SJID)를 포함할 수 있다. 커서에서 NSID와 SJID의 조합은 특정 네임스페이스에 대한 저널(1960)의 리비전 번호를 나타낼 수 있다. 클라이언트 디바이스(150)는 커서를 사용하여 클라이언트 디바이스(150)로 획득된 리비전을 식별할 수 있고, 클라이언트 디바이스(150)로 획득된 리비전에 대응하는 저널(1960)에서의 클라이언트 디바이스(150)의 위치를 식별할 수 있다. 클라이언트 디바이스(150)는 또한 최신 커서를 파일 저널 인터페이스(1902)에 제공하여 저널(1960)에서의 클라이언트 디바이스(150)의 현재 위치를 파일 저널 인터페이스(1902)에 보고할 수 있다. 예를 들어, 클라이언트 디바이스(150)는 클라이언트 디바이스(150)가 새로운 리비전을 요구하는지 여부를 결정하기 위해 커서를 파일 저널 인터페이스(1902)에 제공할 수 있다.

[0329] 클라이언트 디바이스(150)는 또한 클라이언트 디바이스(150)에서의 동작을 파일 저널 인터페이스(1902)에 보고할 때 커서를 파일 저널 인터페이스(1902)에 제공할 수 있다. 커서는 동작을 저널(1960)에서의 특정 리비전 및/또는 저널(1960)에서의 위치에 매핑한다. 이것은 파일 저널 인터페이스(1902)가 클라이언트 디바이스(150)로부터의 동작이 동작으로 수정되는 콘텐츠 아이템에 대한 최신 리비전에 기초하는지 여부를 결정할 수 있도록 한다.

[0330] 클라이언트 디바이스(150)는 커서 및 선형화된 동작 세트를 수신하고, 동작에 기초하여 클라이언트 디바이스(150)에서 콘텐츠 아이템을 업데이트할 수 있다. 이러한 방식으로, 클라이언트 디바이스(150)는 클라이언트 디바이스(150)와 콘텐츠 관리 시스템(110) 사이에 콘텐츠 아이템을 동기화할 수 있다. 클라이언트 디바이스(150)는 저널(1960)에서의 위치를 파일 저널 인터페이스(1902)에 제공하기 위해 커서를 저장할 수 있다.

[0331] 도 21은 네임스페이스 간 동작의 예시적인 선형화를 도시한다. 네임스페이스 간 선형화 및 샤드 간(cross-shard) 또는 네임스페이스 간 리스트는 클록 순서화를 통해 수행할 수 있다. 테이블(2102A, 2102B)(집합적으로 "2102")은 선형화를 위한 네임스페이스 간 동작의 일괄 배치를 도시한다. 테이블(2102A, 2102B)은 각각 테이블(2102A, 2102B) 내의 레코드에 대한 네임스페이스를 식별하기 위한 네임스페이스(NSID) 필드인 열(2106A, 2108A)을 포함하고, 열(2106B, 2108B)은 열(2106A, 2108A) 내의 각 네임스페이스에 대한 테이블(2102A, 2102B)에서의 행 또는 SJID를 식별하기 위한 SJID 필드이고, 열(2106C, 2108C)은 각각의 SJID와 연관된 동작을 식별

하기 위한 동작 필드이고, 열(2106D, 2108D)은 열(2106C, 2108C)에서의 동작과 연관된 타임스탬프를 식별하기 위한 클록 필드이다.

[0332] 이 예에서, 테이블(2102A)은 NSID "1"에 대한 SJID "100" 및 "101"을 도시한다. SJID "100"은 타임스탬프 "1000"에서 네임스페이스 "1"에 "foo.txt"를 추가하는 동작과 연관되고, SJID "101"은 타임스탬프 "1001"에서 네임스페이스 "2"를 마운트하는 동작과 연관된다. 테이블(2102B)은 NSID "2"에 대한 SJID "1" 및 "2"를 도시한다. SJID "1"은 타임스탬프 "2100"에서 네임스페이스 "2"에 "bar.txt"를 추가하는 동작과 연관되고, SJID "2"는 타임스탬프 "1002"에서 "bar.txt"를 편집하는 동작과 연관된다.

[0333] 선형화기(예를 들어, 변환 서비스(1904))는 테이블(2102A 및 2102B)(집합적으로 2102)에서 동작의 일괄 배치를 획득하고, 커서(2114)로 단일 동작 스트림(2112)을 방출할 수 있다. 선형화기는 테이블(2102)에서 적어도 하나의 동작을 갖는 모든 네임스페이스를 식별하고, 각각의 타임스탬프, NSID, SJID에 기초하여 모든 네임스페이스에 대한 동작을 선형화할 수 있다. 이 예에서, 테이블(2102)에서 동작의 일괄 배치는 테이블(2104)에 도시된 동작 스트림으로 선형화된다.

[0334] 테이블(2104)은 각각의 동작의 네임스페이스를 식별하기 위한 NSID 필드를 포함하는 NSID 열(2110), 테이블(2104)에서의 동작을 식별하기 위한 동작 필드를 포함하는 동작 열(2112), 및 각 동작에 대해 커서 상태를 식별하기 위한 커서 필드를 포함하는 커서 열(2114)을 포함한다. 테이블(2104)에서의 행(2104A)은 테이블(2102A)에서의 네임스페이스 "1"의 SJID "100"으로부터의 추가 동작을 포함한다. 행(2104A)에 대한 커서 열(2114)에서의 커서 상태는 네임스페이스 "1" 및 SJID "100"인데, 이것은 추가 동작이 테이블(2102A)에 표시된 네임스페이스 "1"에서의 SJID "100"에 대응하는 것임을 나타낸다. 테이블(2104)에서의 행(2104B)은 NSID 열(2110) 또는 동작 열(2112)에 값을 포함하지는 않지만, 커서 열(2114)에서의 커서 상태를 업데이트하여 네임스페이스 간 커서 상태를 포함하도록 하는데, 이 예에서는 네임스페이스 "2"에 대해 SJID "0"을 추가한다.

[0335] 테이블(2104)에서의 행(2104C)은 테이블(2102A)에 도시된 네임스페이스 "2"에서의 SJID "1"로부터의 추가 동작을 포함한다. 행(2104C)에 대한 커서 열(2114)에서의 커서 상태는 행(2104C)에서의 추가 동작과 연관된 네임스페이스 "1" 및 "2"에 대한 각각의 SJID "100" 및 "1"을 포함한다. 도시된 바와 같이 커서 상태는 커서가 네임스페이스 "1"에서는 SJID "100"에 있고, 네임스페이스 "2"에서는 SJID "1"에 있다는 것을 나타낸다. 즉, 추가 동작이 네임스페이스 "1"의 상태에서는 영향을 미치지 않기 때문에 네임스페이스 "1"에서의 행 또는 SJID는 증가하지 않았지만, 추가 동작이 네임스페이스 "2"에서는 리비전을 나타내고 네임스페이스 "2"의 상태에 영향을 미치기 때문에 네임스페이스 "2"에서의 행 또는 SJID는 1만큼 증가되었다. 따라서, 행(2104C)에서의 커서 상태는 네임스페이스 "2"에서는 SJID "1"에서의 추가 동작 후에 네임스페이스 "1" 및 네임스페이스 "2"에 대한 각각의 SJID를 추적한다.

[0336] 테이블(2104)에서의 행(2104D)은 테이블(2102A)에서의 SJID "101" 및 네임스페이스 "1"에서의 마운트 동작을 포함한다. 마운트 동작은 네임스페이스 "1"에서 네임스페이스 "2"를 마운트한다. 마운트 동작은 네임스페이스 "1"에서의 SJID를 "100"으로부터 "101"로 증가시키지만, 네임스페이스 "2"에서의 SJID를 증가시키지는 않는다. 따라서, 행(2104D)에 대한 커서 열(2114)에서의 커서 상태는 네임스페이스 "1"에 대한 SJID "101"을 포함하고, 네임스페이스 "2"에 대한 SJID "1"을 유지한다. 이 커서 상태는 네임스페이스 "1" 및 "2"의 상태 및/또는 순서를 반영한다.

[0337] 테이블(2104)에서의 행(2104E)은 테이블(2102A)에서 SJID "2" 및 네임스페이스 "2"에서의 편집 동작을 포함하며, 이는 마운트 및 편집 동작의 각각의 타임스탬프에 따라 네임스페이스 "1"에서 SJID "101"에서의 마운트 동작 이후이다. 행(2104E)의 커서 열(2114)에서의 커서 상태는 SJID "101"에서 네임스페이스 "1"에 대한 커서 상태를 유지하지만, 네임스페이스 "2"에 대한 커서 상태를 SJID "2"로 증가시킨다.

[0338] 테이블(2104)에 도시된 바와 같이, 동작(2112)은 네임스페이스 "1" 및 "2"에 걸친 인과 관계 및 타임스탬프에 기초하여 선형화된 동작의 스트림으로서 나열된다. 네임스페이스 간 인과 관계 및 시퀀싱을 반영하기 위해 동작(2112)이 테이블(2104)에서 선형화되면, 동작(2112)은 서버 파일 저널(148)에서의 리비전(예를 들어, 저널(1960)에서의 리비전(1972))으로 변환되고 서버 파일 저널(148)에 기입될 수 있다.

[0339] 예를 들어, 서버 파일 저널(148)에서 네임스페이스 "1"에 대한 저널은 "foo.txt"를 네임스페이스 "1"에 추가하는 추가 동작을 나타내는 SJID "100"에서의 리비전, 및 네임스페이스 "1" 상에 네임스페이스 "2"를 마운트하는 마운트 동작을 나타내는 SJID "101"에서의 리비전을 포함하도록 업데이트될 수 있다. 또한, 서버 파일 저널(148)에서 네임스페이스 "2"에 대한 저널은 네임스페이스 "2"에 "bar.txt"를 추가하는 추가 동작을 나타내는

SJID "1"에서의 리비전, 및 네임스페이스 "2" 상의 "bar.txt"를 편집하는 편집 동작을 나타내는 SJID "2"에서의 리비전을 포함하도록 업데이트될 수 있다.

[0340] 램포트 클록

[0341] 도 22는 이벤트에 대해 계산된 램포트 클록에 따라 정렬된 네임스페이스에 걸친 이벤트를 도시한다. 이 예에서는 네임스페이스(NSID 1, NSID 2 및 NSID 3)에 걸쳐 다양한 동작이 실행되었다. 각 네임스페이스는 네임스페이스 내 동작 순서를 결정하기 위해 이 네임스페이스에서의 모든 동작에 대해 SJID를 유지한다. 그러나 네임스페이스의 SJID는 네임스페이스에 걸쳐 동작의 순서 및 인과 관계를 식별하지는 않는다. 따라서, 네임스페이스(NSID 1, 2, 3)에서의 동작에 대해 램포트 클록이 계산되어 인과 관계를 결정하고 동작의 네임스페이스 간 순서를 얻는다.

[0342] NSID 1에서, 동작(2210)은 SJID 1 및 클록 1을 갖는다. NSID 2에서, 동작(2216)은 SJID 1 및 클록 1을 갖는다. NSID에서, 동작(2220)은 SJID 1 및 클록 1을 갖는다. 동작(2210, 2216, 2220)은 다수의 네임스페이스에 걸쳐 있고, 인과 관계를 갖지 않는다. 따라서, 동작(2210, 2216, 2220)은 서로의 클록에 영향을 미치지 않는다.

[0343] 네임스페이스 내의 동작 순서는 네임스페이스에서의 SJID에 기초하여 결정될 수 있다. 동일한 네임스페이스 내에서의 동작에 대한 클록은 단순히 1만큼 증분될 수 있다. 따라서, NSID 1에서의 SJID 2에서, 동작(2212)에 대한 클록은 2로 증분된다.

[0344] NSID 1에서의 동작(2212)은 File1을 NSID 2로 이동시키는 것이다. 따라서, 동작(2212)은 NSID 2에서 File1을 추가하는 동작(2218)을 NSID 2에서 트리거한다. NSID 2에서의 동작(2218)은 상이한 네임스페이스로부터의 다른 동작, 즉 NSID 1로부터의 동작(2212)에 인과 관계로 의존하기 때문에 동작(2218)에 대한 클록은 NSID 1에서의 클록과 NSID 2에서의 클록에 기초하여 계산된다. 알고리즘은 $타깃NS_클록_{t1} = max(소스_NS_{클록}, 타깃NS_클록_{t0}) + 1$로 표현될 수 있다. 따라서, 이 예에서, NSID 2에서의 동작(2218)에 대한 클록은 3(예를 들어, max(2, 1) + 1)이다. 따라서, NSID 2에서의 동작(2218)은 SJID 2 및 클록 3을 갖는다.

[0345] 유사하게, NSID에서의 동작(2216)은 NSID 2로부터 NSID 1로 File2를 이동시키는 것이다. 따라서, 동작(2216)은 NSID 1에서 File2를 추가하는 동작(2222)을 NSID 1에서 트리거한다. 동작(2222)에 대한 클록은 클록 알고리즘에 기초하여 계산되고, 이는 3이다. 따라서, 동작(2222)은 NSID 1에서 SJID 3 및 클록 3을 갖는다.

[0346] NSID 3에서 동작(2223)은 동일한 네임스페이스에서의 동작, 즉 NSID 3에서의 동작(2220)에 인과 관계로 의존한다. 따라서, 동작(2223)에 대한 클록은 NSID 3에서의 동작(2220)의 클록을 증분시킴으로써 계산될 수 있다. 이 예에서, 동작(2223)에 대한 클록은 2이다. NSID 3에서의 동작(2223)은 SJID 2 및 클록 2를 갖는다. 동작(2223)은 Dir을 NSID 1로 이동시키기 위한 이동 동작이므로, 동작(2223)은 Dir을 NSID 1에 추가하는 동작(2224)을 NSID 1에서 트리거한다.

[0347] 동작(2224)은 상이한 네임스페이스(NSID 3)에서 동작(2222)에 의해 트리거되기 때문에, 동작(2224)에 대한 클록은 NSID 1에서의 클록 및 동작(2222)에 대한 클록에 기초하여 계산된다. 따라서, 동작(2224)에 대한 클록은 4(예를 들어, max(2, 3+1))로 설정된다. 따라서, 동작(2224)은 NSID 1에서 SJID 4 및 클록 4를 갖는다.

[0348] NSID 1에서의 동작(2226)은 File3을 NSID 1에 추가하며, 네임스페이스 간 동작이 아니다. 따라서, 동작(2226)에 대한 클록은 NSID 1에서 클록을 증분시킴으로써 계산된다. 따라서, 동작(2226)에 대한 클록은 5로 설정된다.

[0349] 동작(2228)은 NSID 1 내에서도 동작(2226)에 인과 관계로 의존한다. 따라서, 동작(2228)에 대한 클록은 NSID 1에서 동작(2226)의 클록을 증분시킴으로써 6으로 설정된다. 동작(2228)은 NSID 1에서 SJID 6 및 클록 6을 갖는다.

[0350] 동작(2228)은 File3을 NSID 3으로 이동시키는 이동 동작이다. 따라서 동작(2228)은 NSID 3에서 동작(2230)을 트리거한다. 동작(2230)은 상이한 네임스페이스로부터의 동작에 기초하기 때문에, 클록은 NSID 3에서의 클록 및 동작(2228)의 클록에 기초하여 클록 알고리즘을 사용하여 계산된다. 이 경우, 동작(2230)에 대한 클록은 7로 설정된다. 따라서 동작(2230)은 NSID 3에서의 SJID 3 및 클록 7을 갖는다.

[0351] 동작(2232, 2234)은 네임스페이스 간 동작이 아니며 NSID 3에서의 동작(2230)과 인과 관계로 관련된다. 따라서, 동작(2232, 2234)에 대한 클록은 동작(2230)의 클록을 증분시킴으로써 계산될 수 있다. 이 예에서, 동작(2232, 2234)에 대한 클록은 각각 8 및 9로 설정된다.

[0352] 도 23은 예를 들어 연결(2305)을 사용하여 서로 통신하는 클라이언트 디바이스(150), 콘텐츠 관리 시스템(110),

또는 임의의 구성 요소를 구성하는 임의의 컴퓨팅 디바이스일 수 있는 컴퓨팅 시스템(2300)의 일례를 도시한다. 연결(2305)은 버스를 통한 물리적 연결이거나, 또는 칩셋 아키텍처와 같이 프로세서(2310)로의 직접 연결일 수 있다. 연결(2305)은 가상 연결, 네트워크 연결 또는 논리적 연결일 수도 있다.

[0353] 일부 실시형태에서, 컴퓨팅 시스템(2300)은 본 명세서에서 설명된 기능이 데이터 센터, 다수의 데이터 센터, 피어 네트워크 등으로 분산될 수 있는 분산 시스템이다. 일부 실시형태에서, 설명된 시스템 구성 요소 중 하나 이상은 설명된 구성 요소의 기능의 일부 또는 전부를 각각 수행하는 만큼 많은 구성 요소를 나타낸다. 일부 실시형태에서, 구성 요소는 물리적 디바이스이거나 또는 가상 디바이스일 수 있다.

[0354] 예시적인 시스템(2300)은 적어도 하나의 처리 유닛(CPU 또는 프로세서)(2310), 및 판독 전용 메모리(ROM)(2320) 및 랜덤 액세스 메모리(RAM)(2325)와 같은 시스템 메모리(2315)를 포함하는 다양한 시스템 구성 요소들을 프로세서(2310)에 결합시키는 연결(2305)을 포함한다. 컴퓨팅 시스템(2300)은 프로세서(2310)의 일부와 직접 연결되는, 그 일부에 근접하는, 또는 그 일부와 통합된 고속 메모리(2312)의 캐시를 포함할 수 있다.

[0355] 프로세서(2310)는 프로세서(2310)를 제어하도록 구성된 저장 디바이스(2330)에 저장된 서비스(2332, 2334 및 2336)와 같은 임의의 범용 프로세서 및 하드웨어 서비스 또는 소프트웨어 서비스, 및 소프트웨어 명령이 실제 프로세서 설계에 통합된 특수 목적 프로세서를 포함할 수 있다. 프로세서(2310)는 본질적으로 다수의 코어 또는 프로세서, 버스, 메모리 제어기, 캐시 등을 포함하는 완전히 독립적인 컴퓨팅 시스템일 수 있다. 멀티-코어 프로세서는 대칭 또는 비대칭일 수 있다.

[0356] 사용자 상호 작용을 가능하게 하기 위해, 컴퓨팅 시스템(2300)은 음성용 마이크로폰, 제스처 또는 그래픽 입력용 터치 감지 스크린, 키보드, 마우스, 모션 입력, 음성 등과 같은 임의의 수의 입력 메커니즘을 나타낼 수 있는 입력 디바이스(2345)를 포함한다. 컴퓨팅 시스템(2300)은 또한 이 기술 분야에 통상의 지식을 가진 자에게 알려진 다수의 출력 메커니즘 중 하나 이상일 수 있는 출력 디바이스(2335)를 포함할 수 있다. 일부 경우에, 멀티모달 시스템은 사용자가 컴퓨팅 시스템(2300)과 통신하기 위해 다수의 유형의 입력/출력을 제공할 수 있게 한다. 컴퓨팅 시스템(2300)은 일반적으로 사용자 입력 및 시스템 출력을 제어 및 관리할 수 있는 통신 인터페이스(2340)를 포함할 수 있다. 임의의 특정 하드웨어 배열에서 동작하는 데 제한이 있는 것은 아니므로 본 발명에서 기본 기능은 개선된 하드웨어 또는 펌웨어 배열이 개발될 때 쉽게 대체될 수 있다.

[0357] 저장 디바이스(2330)는 비-휘발성 메모리 디바이스일 수 있고, 컴퓨터에 의해 액세스 가능한 데이터를 저장할 수 있는 하드 디스크 또는 다른 유형의 컴퓨터 판독 가능 매체일 수 있고, 예를 들어, 자기 카세트, 플래시 메모리 카드, 솔리드 스테이트 메모리 디바이스, 디지털 다용도 디스크, 카트리지, 랜덤 액세스 메모리(RAM), 판독 전용 메모리(ROM) 및/또는 이들 디바이스의 일부 조합일 수 있다.

[0358] 저장 디바이스(2330)는, 소프트웨어를 정의하는 코드가 프로세서(2310)에 의해 실행될 때, 시스템으로 하여금 기능을 수행하게 하는 소프트웨어 서비스, 서버, 서비스 등을 포함할 수 있다. 일부 실시형태에서, 특정 기능을 수행하는 하드웨어 서비스는 기능을 수행하도록 프로세서(2310), 연결(2305), 출력 디바이스(2335) 등과 같은 필요한 하드웨어 구성 요소와 관련하여 컴퓨터 판독 가능 매체에 저장된 소프트웨어 구성 요소를 포함할 수 있다.

[0359] 설명의 명료함을 위해, 일부 예에서, 본 기술은 디바이스, 디바이스 구성 요소, 소프트웨어로 구현된 방법의 단계 또는 루틴, 또는 하드웨어 및 소프트웨어의 조합을 포함하는 기능 블록을 포함하는 개별 기능 블록을 포함하는 것으로 제시될 수 있다.

[0360] 본 명세서에 기술된 단계, 동작, 기능 또는 프로세스 중 임의의 것은 하드웨어와 소프트웨어 서비스의 조합에 의해 또는 단독으로 또는 다른 디바이스와 조합하여 서비스에 의해 수행되거나 구현될 수 있다. 일부 실시형태에서, 서비스는 클라이언트 디바이스의 메모리에 상주하고/하거나 콘텐츠 관리 시스템의 하나 이상의 서버에 상주하며 프로세서가 서비스와 연관된 소프트웨어를 실행할 때 하나 이상의 기능을 수행하는 소프트웨어일 수 있다. 일부 실시형태에서, 서비스는 특정 기능을 수행하는 프로그램 또는 프로그램의 집합이다. 일부 실시형태에서, 서비스는 서버로 간주될 수 있다. 메모리는 비-일시적인 컴퓨터 판독 가능 매체일 수 있다.

[0361] 일부 실시형태에서, 컴퓨터 판독 가능 저장 디바이스, 매체 및 메모리는 비트 스트림 등을 포함하는 무선 신호 또는 케이블을 포함할 수 있다. 그러나, 언급될 때, 비-일시적인 컴퓨터 판독 가능 저장 매체는 에너지, 캐리어 신호, 전자기파 및 신호 자체와 같은 매체를 명시적으로 배제한다.

[0362] 전술한 예에 따른 방법은 컴퓨터 판독 가능 매체로부터 저장되거나 달리 이용 가능한 컴퓨터 실행 가능 명령을 사용하여 구현될 수 있다. 이러한 명령은 예를 들어, 범용 컴퓨터, 특수 목적 컴퓨터 또는 특정 목적 처리 디바

이스로 하여금 특정 기능 또는 기능 그룹을 수행하게 하거나 달리 특정 기능 또는 기능 그룹을 수행하도록 컴퓨터 또는 디바이스를 구성하게 하는 명령 및 데이터를 포함할 수 있다. 사용된 컴퓨터 자원의 일부는 네트워크를 통해 액세스될 수 있다. 컴퓨터 실행 가능 명령은 예를 들어, 바이너리, 어셈블리 언어와 같은 중간 포맷 명령, 펌웨어 또는 소스 코드(source code)일 수 있다. 설명된 예에 따른 방법 동안 명령, 사용된 정보 및/또는 생성된 정보를 저장하는데 사용될 수 있는 컴퓨터 판독 가능 매체의 예로는 자기 또는 광 디스크, 솔리드 스테이트 메모리 디바이스, 플래시 메모리, 비-휘발성 메모리가 제공된 USB 디바이스, 네트워크 저장 디바이스 등을 포함한다.

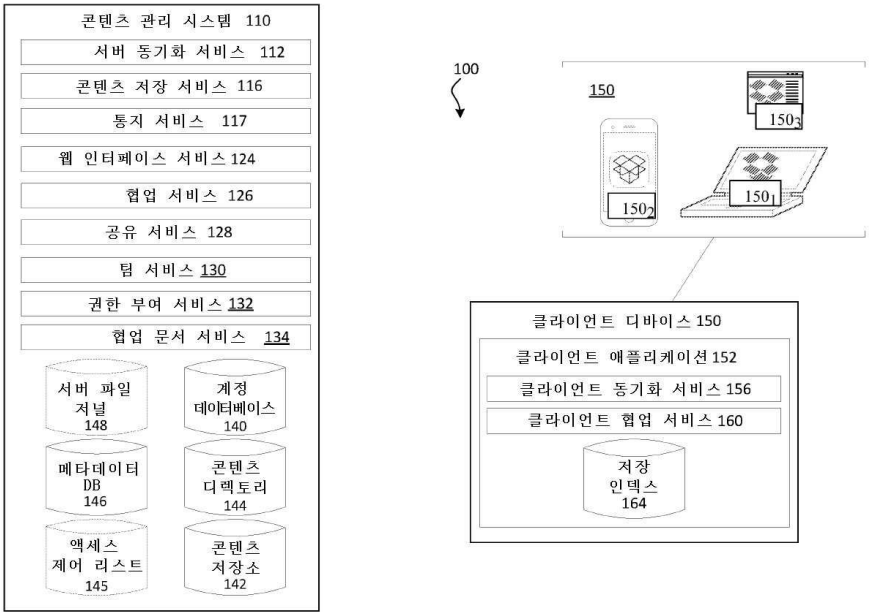
[0363] 본 발명에 따른 방법을 구현하는 디바이스는 하드웨어, 펌웨어 및/또는 소프트웨어를 포함할 수 있고, 다양한 폼 팩터 중 임의의 것을 취할 수 있다. 이러한 폼 팩터의 일반적인 예로는 서버, 랩톱, 스마트 폰, 소형 폼 팩터 개인용 컴퓨터, 개인 휴대 정보 단말기 등을 포함한다. 본 명세서에 기술된 기능은 또한 주변 장치 또는 애드인 카드에 구현될 수 있다. 이러한 기능은 또한 다른 예로서, 단일 디바이스에서 실행되는 상이한 프로세스 또는 상이한 칩들 중 회로 보드에서 구현될 수 있다.

[0364] 명령, 이러한 명령을 전달하기 위한 매체, 이 명령을 실행하기 위한 컴퓨팅 자원, 및 이러한 컴퓨팅 자원을 지원하기 위한 다른 구조는 본 명세서에 기술된 기능을 제공하기 위한 수단이다.

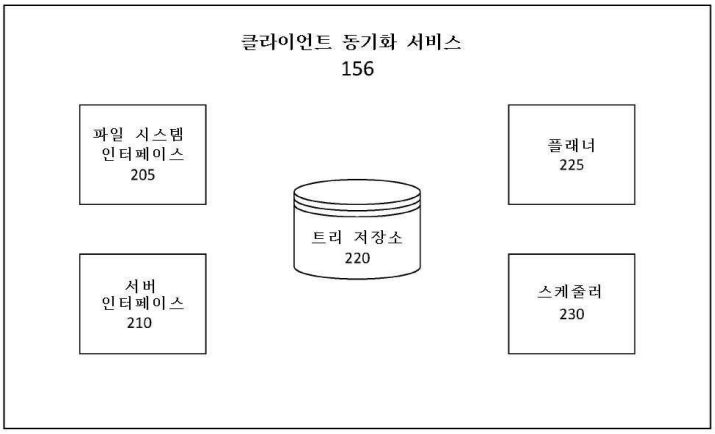
[0365] 첨부된 청구범위 내에서 양태를 설명하기 위해 다양한 예 및 다른 정보가 사용되었지만, 이 기술 분야에 통상의 지식을 가진 자라면 이 예를 사용하여 다양한 구현을 도출할 수 있으므로, 이러한 예의 특정 특징 또는 배열에 기초하여 청구범위를 제한하는 것으로 이해되어서는 안 된다. 또한, 일부 주제는 구조적 특징 및/또는 방법 단계의 예에 특정한 언어로 설명되었지만, 첨부된 청구범위에 한정된 주제는 이러한 설명된 특징 또는 행위로 반드시 제한되는 것은 아니라는 것을 이해해야 한다. 예를 들어, 이러한 기능은 본 명세서에서 식별된 것과 다른 구성 요소로 상이하게 분산되어 수행될 수 있다. 오히려, 설명된 특징 및 단계는 첨부된 청구범위 내에서 시스템 및 방법의 구성 요소의 예로서 개시된다.

도면

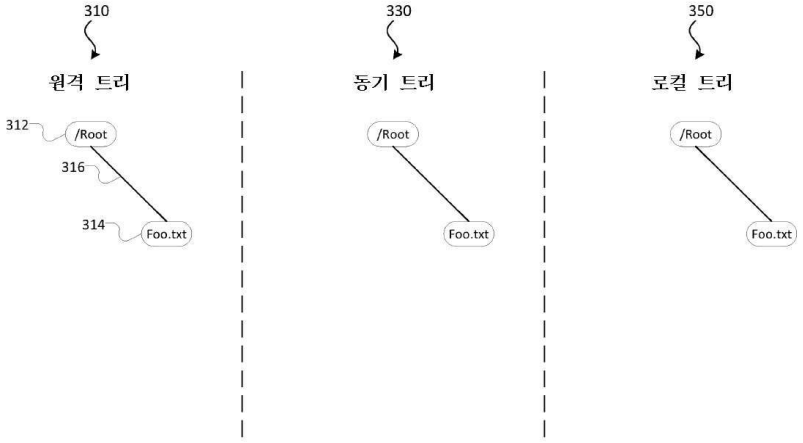
도면1



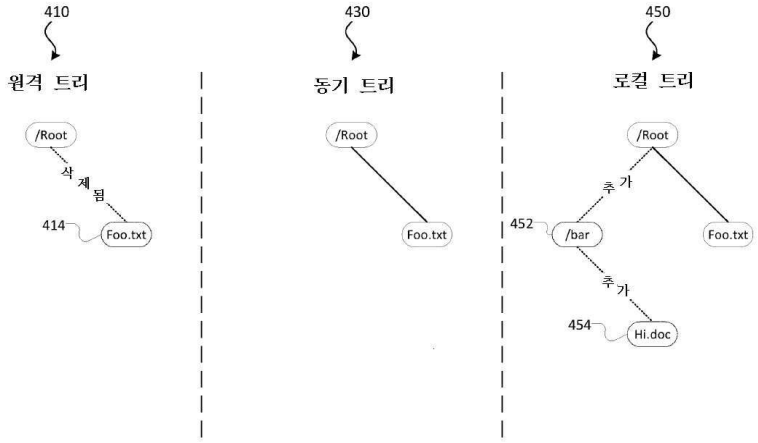
도면2


클라이언트 동기화 서비스
156
파일 시스템
인터페이스
205
트리 저장소
220
플래너
225
서버
인터페이스
210
스케줄러
230

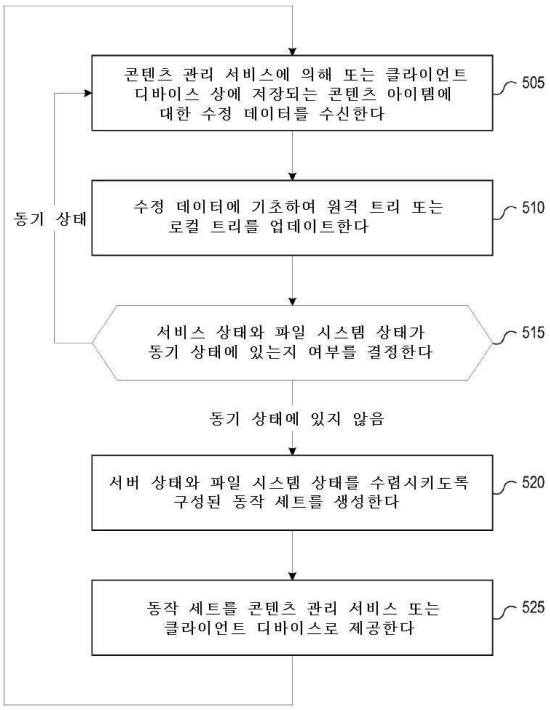
도면3


310
원격 트리
312
/Root
316
314
Foo.txt
330
동기 트리
/Root
Foo.txt
350
로컬 트리
/Root
Foo.txt

도면4

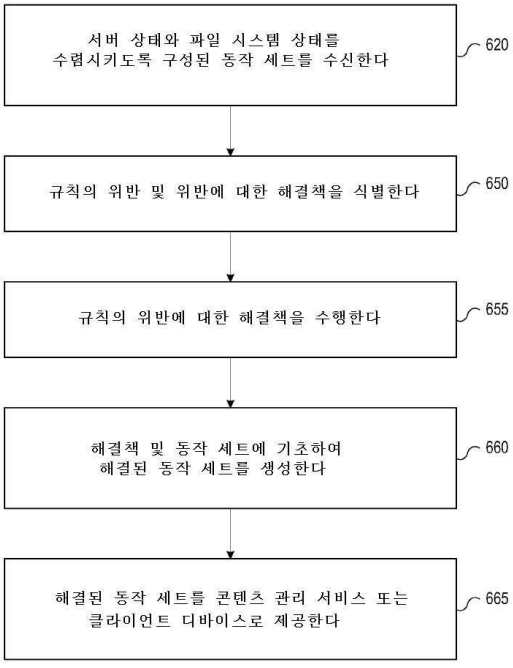

410
원격 트리
/Root
삭제됨
414
Foo.txt
430
동기 트리
/Root
Foo.txt
450
로컬 트리
/Root
추가
452
/bar
Foo.txt
추가
454
Hi.doc

도면5



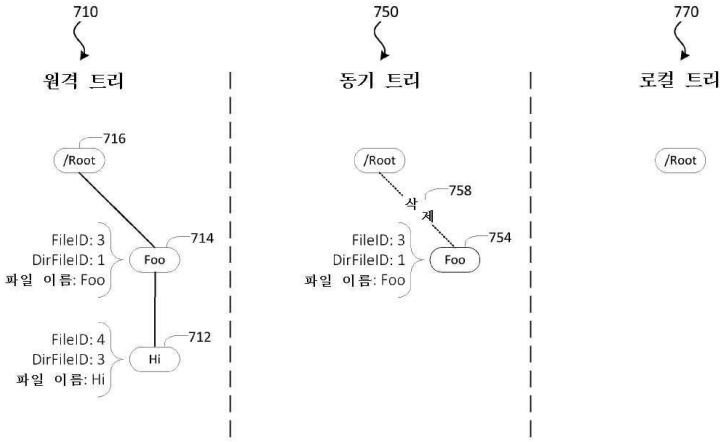
500

도면6

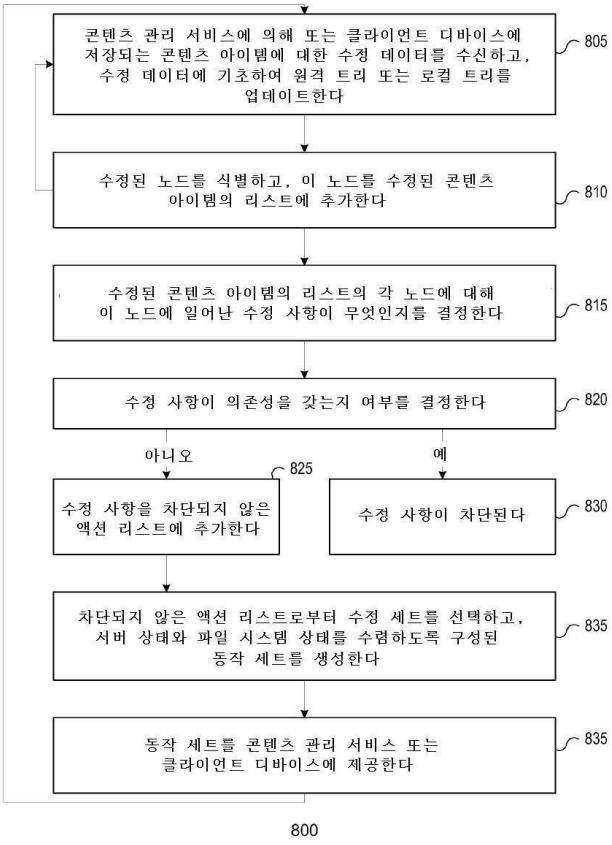


600

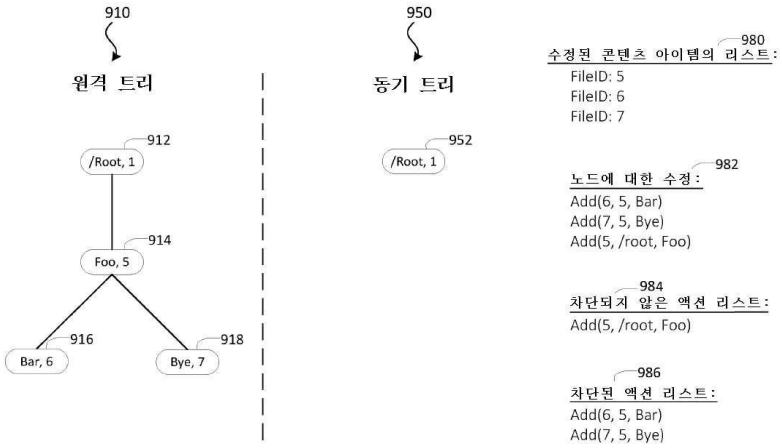
도면7


710
원격 트리
750
동기 트리
770
로컬 트리
716
/Root
/Root
/Root
758
삭
제
FileID: 3
DirFileID: 1
파일 이름: Foo
714
Foo
754
Foo
FileID: 4
DirFileID: 3
파일 이름: Hi
712
Hi

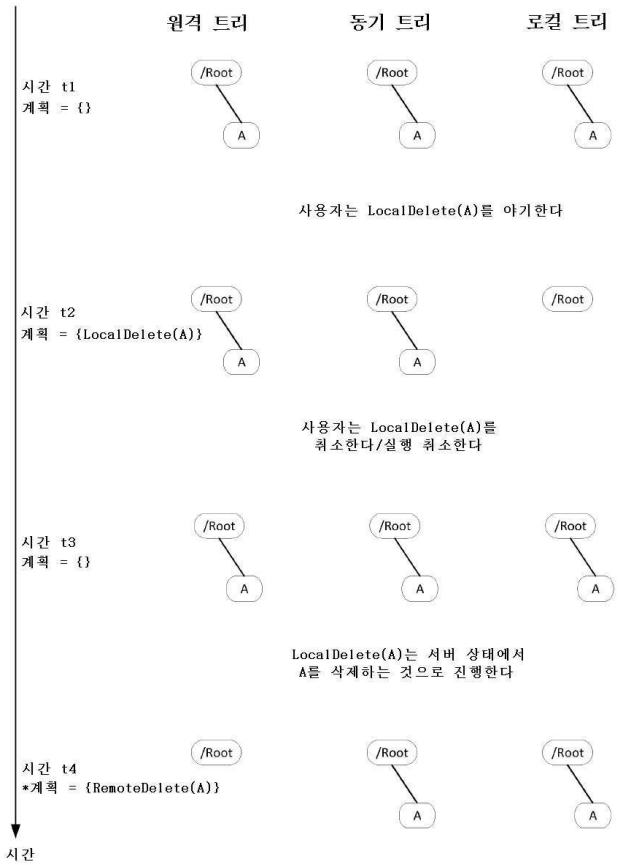
도면8


콘텐츠 관리 서비스에 의해 또는 클라이언트 디바이스에 저장되는 콘텐츠 아이템에 대한 수정 데이터를 수신하고, 수정 데이터에 기초하여 원격 트리 또는 로컬 트리를 업데이트한다
805
수정된 노드를 식별하고, 이 노드를 수정된 콘텐츠 아이템의 리스트에 추가한다
810
수정된 콘텐츠 아이템의 리스트의 각 노드에 대해 이 노드에 일어난 수정 사항이 무엇인지를 결정한다
815
수정 사항이 의존성을 갖는지 여부를 결정한다
820
아니오
예
825
수정 사항을 차단되지 않은 액션 리스트에 추가한다
수정 사항이 차단된다
830
차단되지 않은 액션 리스트로부터 수정 세트를 선택하고, 서버 상태와 파일 시스템 상태를 수렴하도록 구성된 동작 세트를 생성한다
835
동작 세트를 콘텐츠 관리 서비스 또는 클라이언트 디바이스에 제공한다
835
800

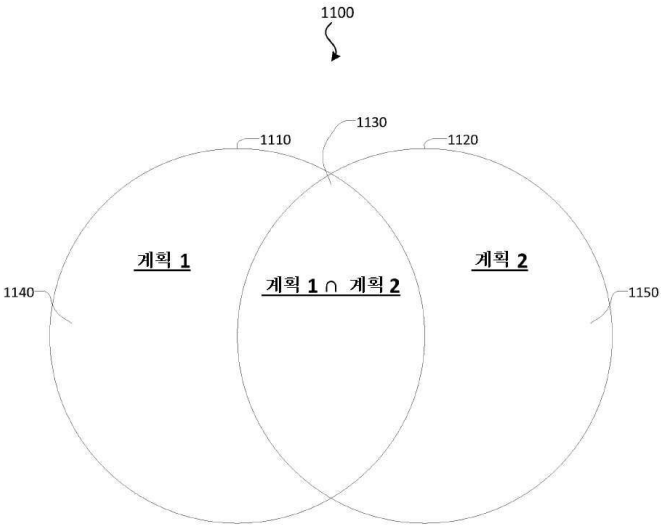
도면9


910
원격 트리
912
/Root, 1
914
Foo, 5
916
Bar, 6
918
Bye, 7
950
동기 트리
952
/Root, 1
980
수정된 콘텐츠 아이템의 리스트:
FileID: 5
FileID: 6
FileID: 7
982
노드에 대한 수정:
Add(6, 5, Bar)
Add(7, 5, Bye)
Add(5, /root, Foo)
984
차단되지 않은 액션 리스트:
Add(5, /root, Foo)
986
차단된 액션 리스트:
Add(6, 5, Bar)
Add(7, 5, Bye)

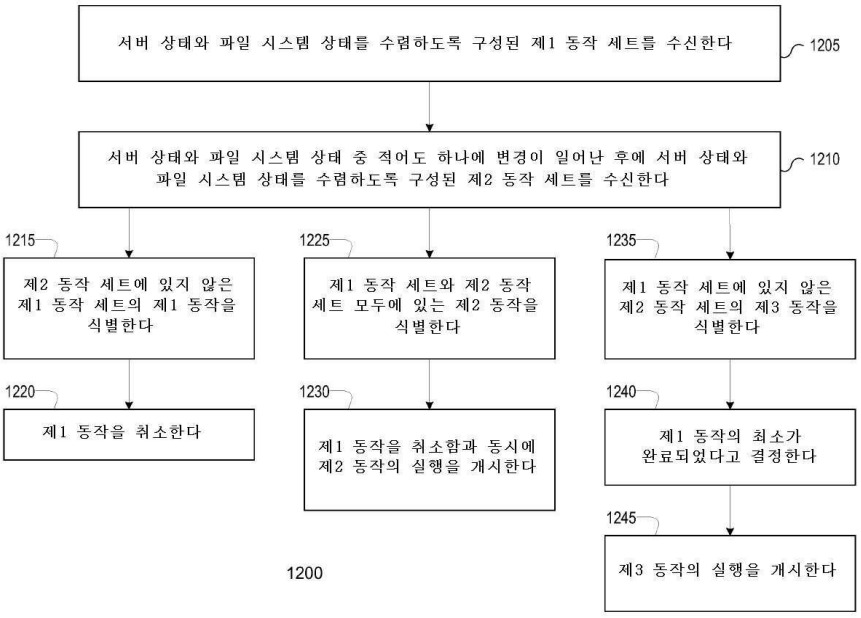
도면10


원격 트리
동기 트리
로컬 트리
시간 t1
계획 = {}
/Root
A
사용자는 LocalDelete(A)를 아기한다
시간 t2
계획 = {LocalDelete(A)}
사용자는 LocalDelete(A)를
취소한다/실행 취소한다
시간 t3
계획 = {}
LocalDelete(A)는 서버 상태에서
A를 삭제하는 것으로 진행한다
시간 t4
*계획 = {RemoteDelete(A)}
시간

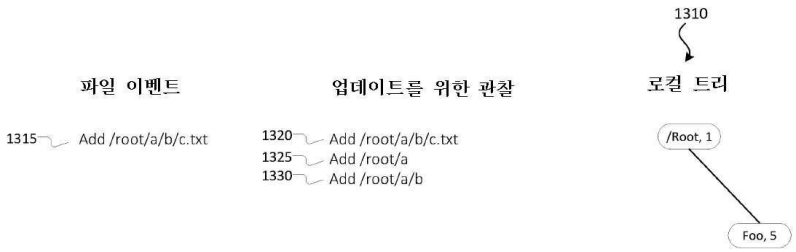
도면11



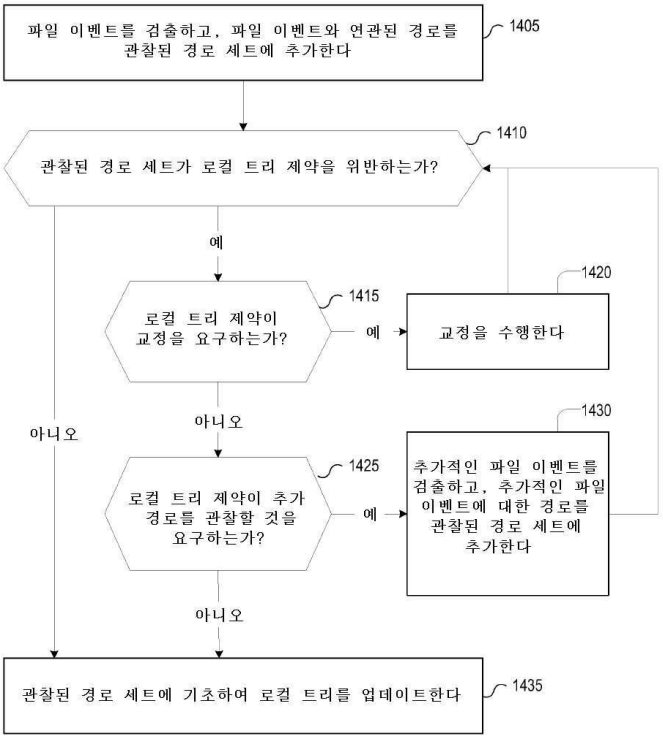
도면12



도면13

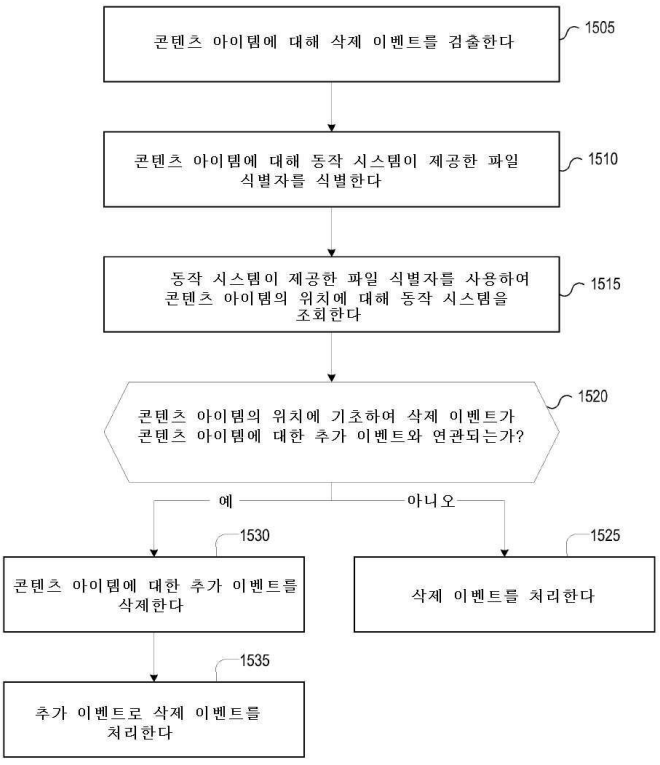


도면14



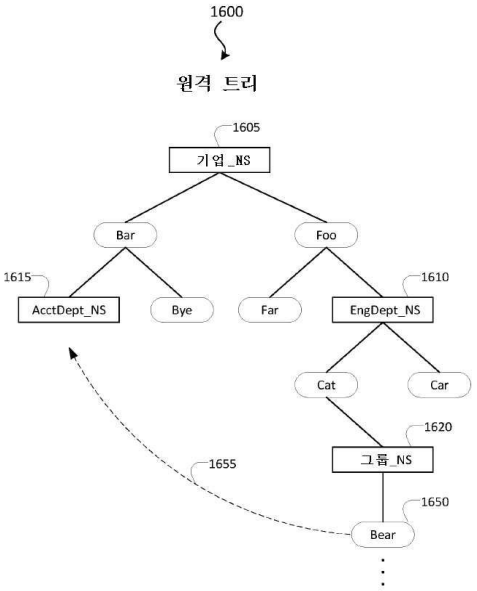
1400

도면15

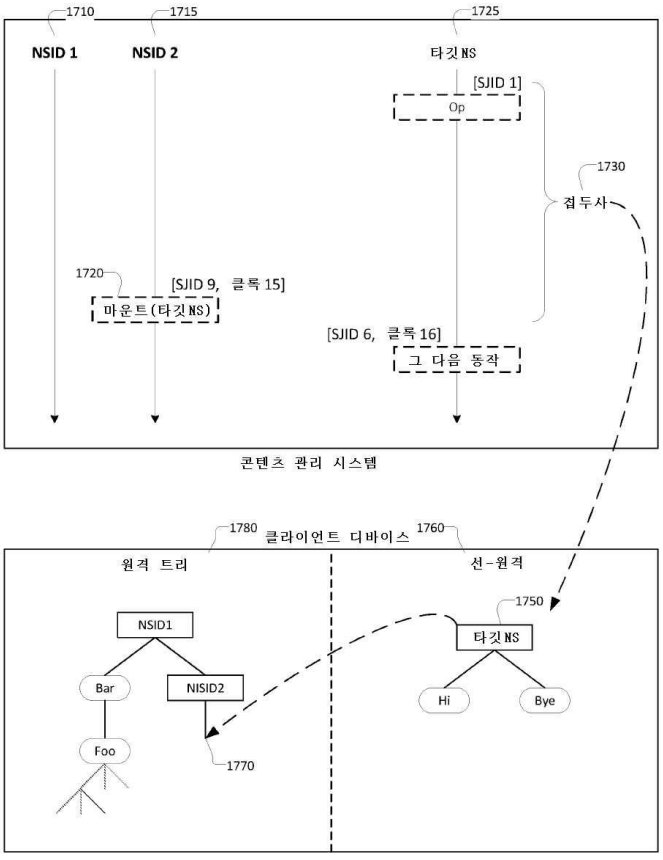


1500

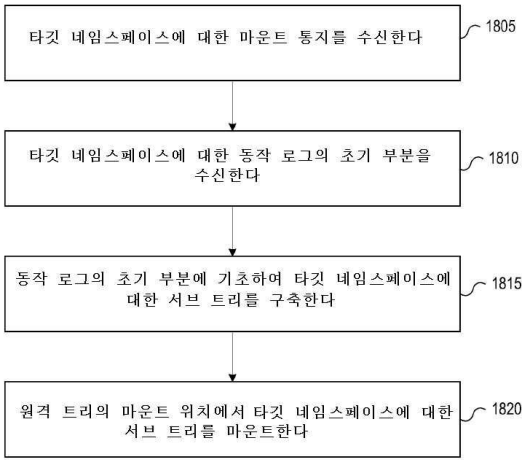
도면16


1600
원격 트리
1605
기업_NS
Bar
Foo
1615
1610
AcctDept_NS
Bye
Far
EngDept_NS
Cat
Car
1620
그룹_NS
1655
1650
Bear

도면17

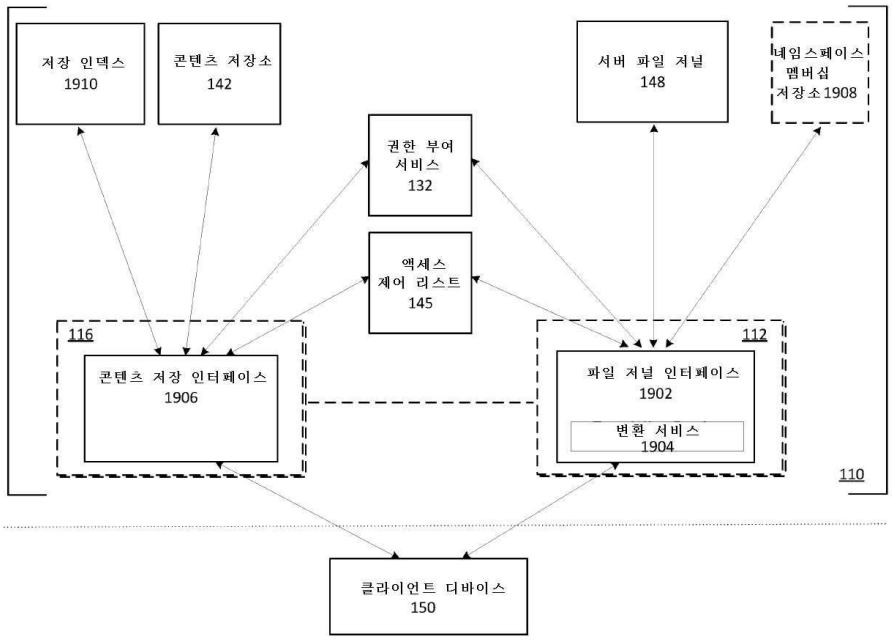

1710
1715
1725
NSID 1
NSID 2
타깃 NS
[SJID 1]
Op
1730
접두사
1720
[SJID 9, 클록 15]
마운트(타깃 NS)
[SJID 6, 클록 16]
그 다음 동작
콘텐츠 관리 시스템
1780
클라이언트 디바이스
1760
원격 트리
선-원격
1750
NSID1
타깃 NS
Bar
NISID2
Hi
Bye
Foo
1770

도면18

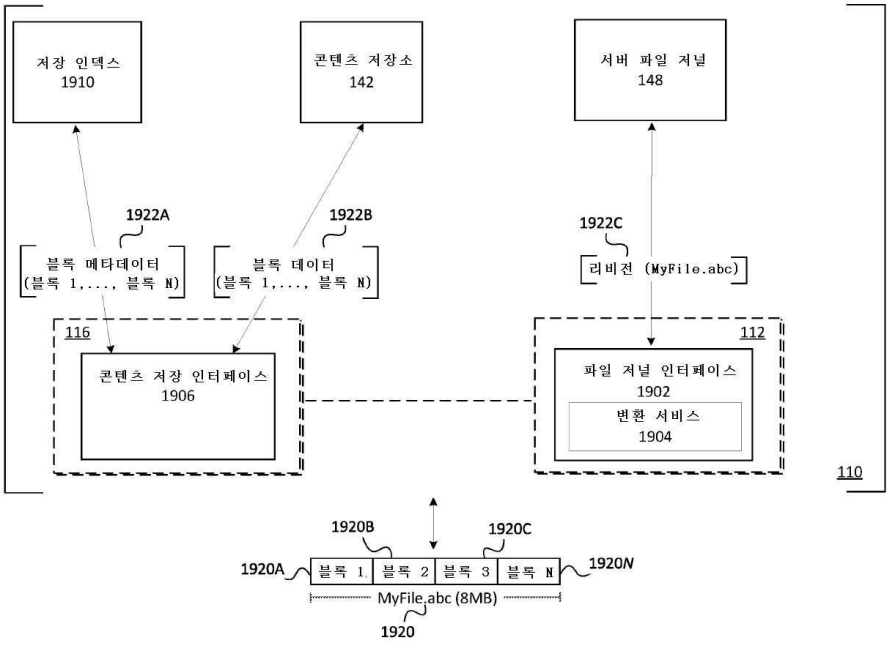


1800

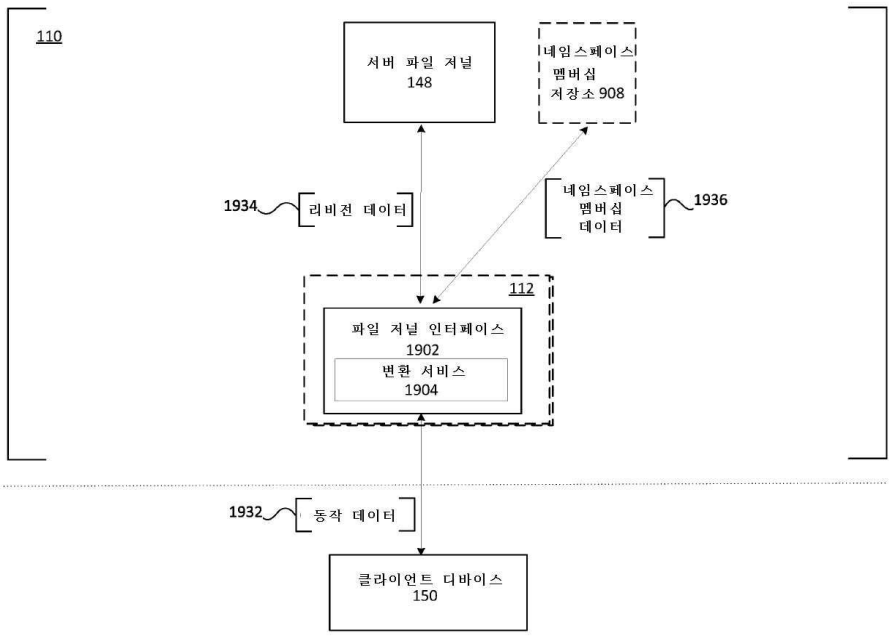
도면19a



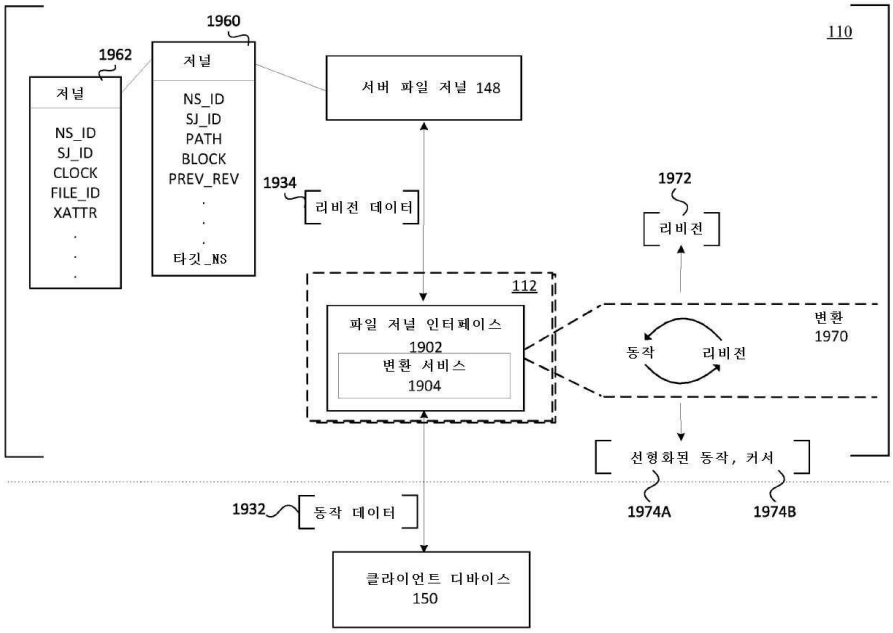
도면19b


저장 인덱스
1910
콘텐츠 저장소
142
서버 파일 저널
148
1922A
블록 메타데이터
(블록 1,..., 블록 N)
1922B
블록 데이터
(블록 1,..., 블록 N)
1922C
리비전 (MyFile.abc)
116
콘텐츠 저장 인터페이스
1906
112
파일 저널 인터페이스
1902
변환 서비스
1904
110
1920B
1920C
1920A
블록 1
블록 2
블록 3
블록 N
1920N
MyFile.abc (8MB)
1920

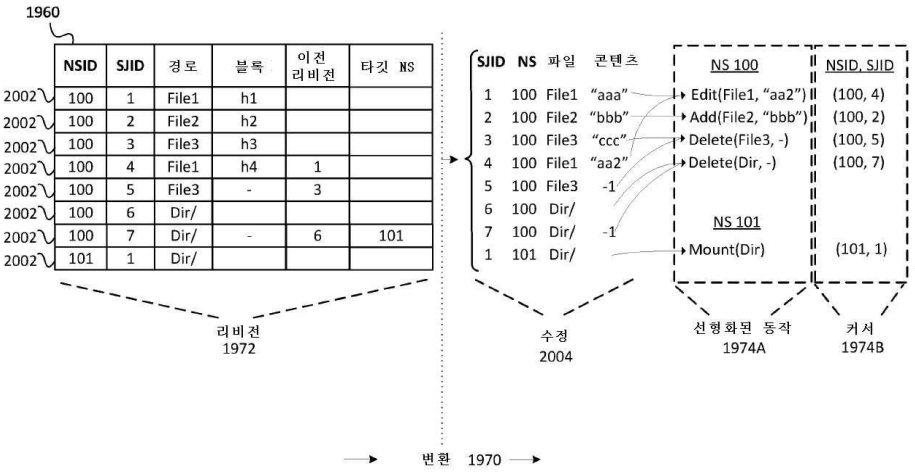
도면19c


110
서버 파일 저널
148
네임스페이스
멤버십
저장소 908
1934
리비전 데이터
네임스페이스
멤버십
데이터
1936
112
파일 저널 인터페이스
1902
변환 서비스
1904
1932
동작 데이터
클라이언트 디바이스
150

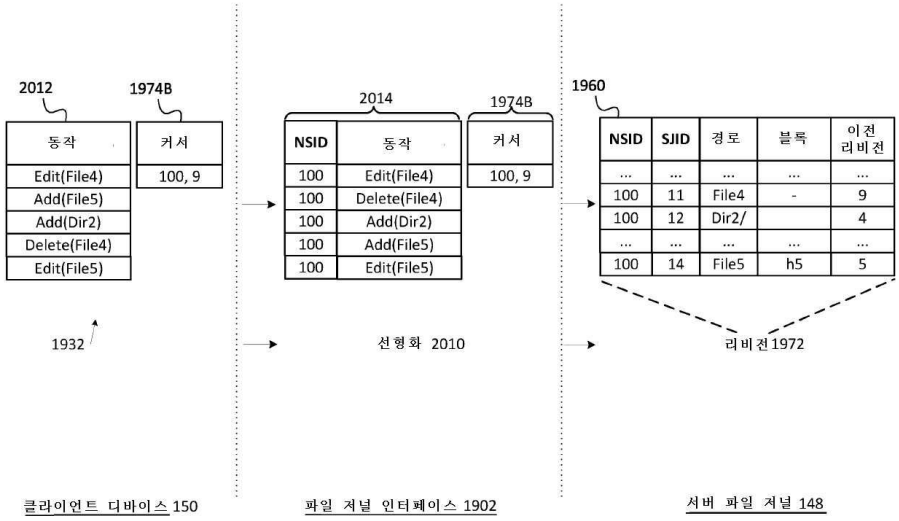
도면19d


110
1960
1962
저널
NS_ID
SJ_ID
CLOCK
FILE_ID
XATTR
저널
NS_ID
SJ_ID
PATH
BLOCK
PREV_REV
타깃_NS
서버 파일 저널 148
1934
리비전 데이터
1972
리비전
112
파일 저널 인터페이스
1902
변환 서비스
1904
변환
1970
동작
리비전
선형화된 동작, 커서
1974A
1974B
1932
동작 데이터
클라이언트 디바이스
150

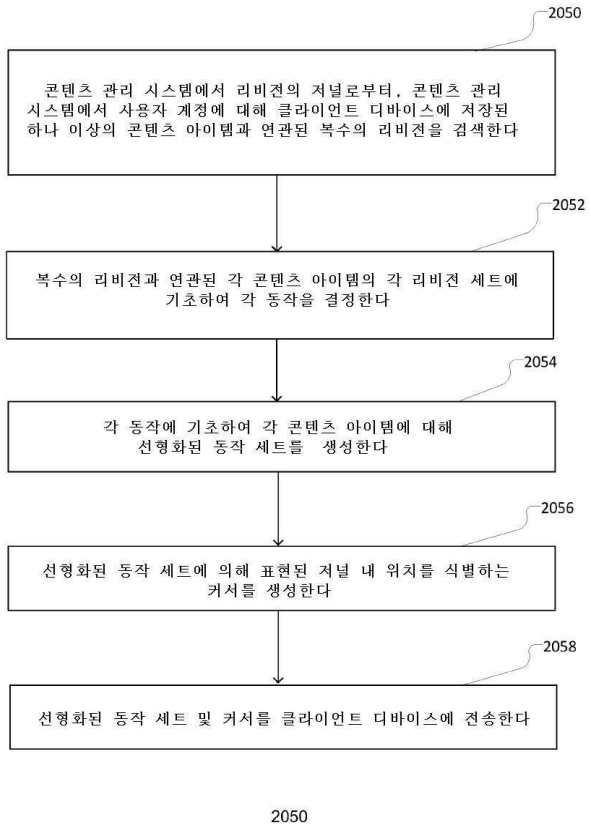
도면20a


1960
NSID
SJID
경로
블록
이전 리비전
타깃 NS
2002
100 1 File1 h1
100 2 File2 h2
100 3 File3 h3
100 4 File1 h4 1
100 5 File3 - 3
100 6 Dir/
100 7 Dir/ - 6 101
101 1 Dir/
리비전
1972
SJID NS 파일 콘텐츠
1 100 File1 "aaa"
2 100 File2 "bbb"
3 100 File3 "ccc"
4 100 File1 "aa2"
5 100 File3 -1
6 100 Dir/
7 100 Dir/ -1
1 101 Dir/
수정
2004
NS 100
Edit(File1, "aa2")
Add(File2, "bbb")
Delete(File3, -)
Delete(Dir, -)
NS 101
Mount(Dir)
NSID, SJID
(100, 4)
(100, 2)
(100, 5)
(100, 7)
(101, 1)
선형화된 동작
1974A
커서
1974B
변환 1970
서버 파일 저널 148
파일 저널 인터페이스 1902

도면20b

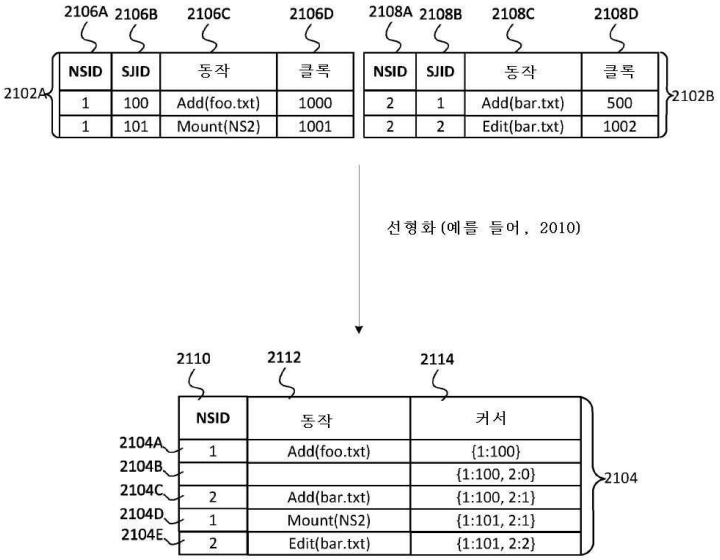


도면20c

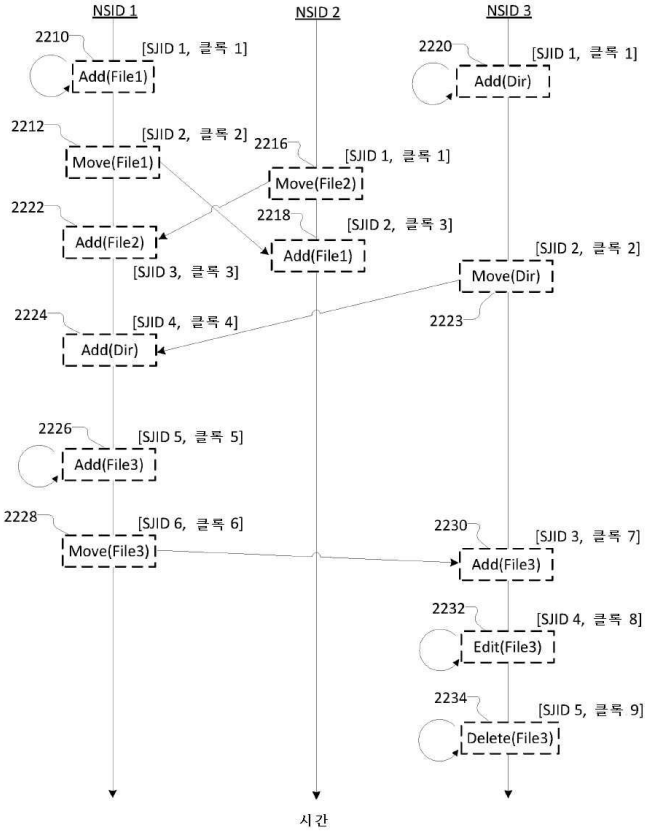


2050

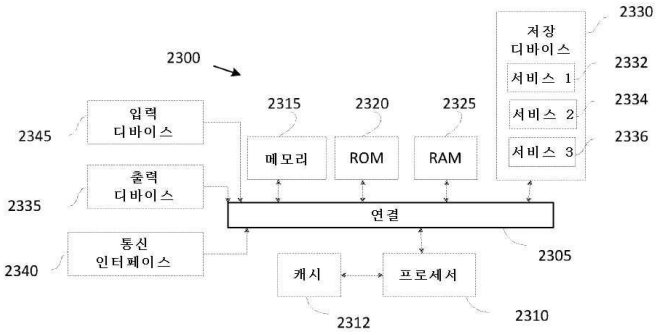
도면21


2106A
2106B
2106C
2106D
2108A
2108B
2108C
2108D
NSID
SJID
동작
클록
2102A
1
100
Add(foo.txt)
1000
1
101
Mount(NS2)
1001
2
1
Add(bar.txt)
500
2
2
Edit(bar.txt)
1002
2102B
선형화 (예를 들어, 2010)
2110
2112
2114
NSID
동작
커서
2104A
2104B
2104C
2104D
2104E
1
Add(foo.txt)
{1:100}
{1:100, 2:0}
2
Add(bar.txt)
{1:100, 2:1}
1
Mount(NS2)
{1:101, 2:1}
2
Edit(bar.txt)
{1:101, 2:2}
2104

도면22


NSID 1
NSID 2
NSID 3
2210
[SJID 1, 클록 1]
Add(File1)
2220
[SJID 1, 클록 1]
Add(Dir)
2212
[SJID 2, 클록 2]
Move(File1)
2216
[SJID 1, 클록 1]
Move(File2)
2222
Add(File2)
[SJID 3, 클록 3]
2218
[SJID 2, 클록 3]
Add(File1)
[SJID 2, 클록 2]
Move(Dir)
2224
[SJID 4, 클록 4]
Add(Dir)
2223
2226
[SJID 5, 클록 5]
Add(File3)
2228
[SJID 6, 클록 6]
Move(File3)
2230
[SJID 3, 클록 7]
Add(File3)
2232
[SJID 4, 클록 8]
Edit(File3)
2234
[SJID 5, 클록 9]
Delete(File3)
시간

도면23


2300
2330
저장
디바이스
2332
서비스 1
2334
서비스 2
2336
서비스 3
2315
2320
2325
입력
디바이스
2345
메모리
ROM
RAM
출력
디바이스
2335
연결
2305
통신
인터페이스
2340
캐시
프로세서
2312
2310