



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2015년01월19일

(11) 등록번호 10-1484333

(24) 등록일자 2015년01월13일

(51) 국제특허분류(Int. Cl.)

H04L 29/06 (2006.01) G06F 17/30 (2006.01)

(21) 출원번호 10-2009-7014101

(22) 출원일자(국제) 2007년08월30일

심사청구일자 2012년08월13일

(85) 번역문제출일자 2009년07월06일

(65) 공개번호 10-2009-0099549

(43) 공개일자 2009년09월22일

(86) 국제출원번호 PCT/US2007/077225

(87) 국제공개번호 WO 2008/076481

국제공개일자 2008년06월26일

(30) 우선권주장

11/640,658 2006년12월18일 미국(US)

(56) 선행기술조사문헌

JP2002094607 A

전체 청구항 수 : 총 20 항

(73) 특허권자

마이크로소프트 코포레이션

미국 워싱턴주 (우편번호 : 98052) 레드몬드 원  
마이크로소프트 웨이

(72) 발명자

리, 올리버

미국 98052-6399 워싱턴주 레드몬드 원 마이크로  
소프트 웨이 마이크로소프트 코포레이션 국제 특  
허부 내

비커맨, 마이클, 스코트

미국 98052-6399 워싱턴주 레드몬드 원 마이크로  
소프트 웨이 마이크로소프트 코포레이션 국제 특  
허부 내

(뒷면에 계속)

(74) 대리인

제일특허법인

심사관 : 이철수

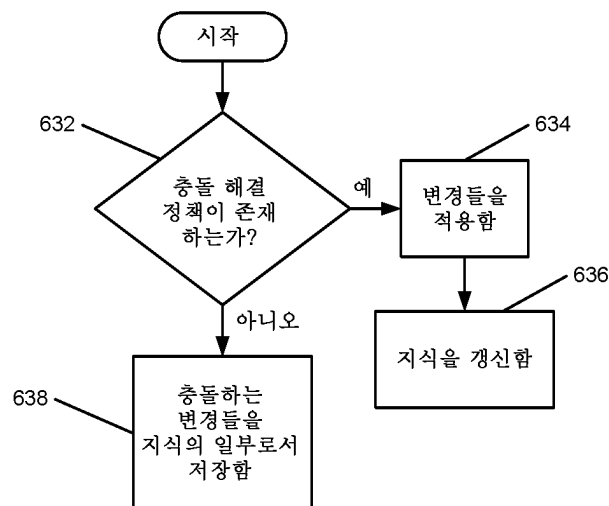
(54) 발명의 명칭 **충돌 지식의 전파**

### (57) 요약

동기화 커뮤니티는 일군의 동기화 중단점들을 포함할 수 있다. 동기화 커뮤니티의 2개의 동기화 중단점이 서로 동기화를 할 때, 2개의 동기화 중단점이 동일한 특성의 데이터 항목을 변경하고 그 변경이 행해졌을 때 2개의 동기화 중단점이 각자의 상대방 동기화 중단점에 의해 행해진 그 동일한 특성의 데이터에 대한 변경을 알아채지 못한 경우 동기화 데이터 충돌이 검출될 수 있다. 검출된 동기화 데이터 충돌의 해결이 지연될 수 있고, 검출된 동기화 데이터 충돌을 나타내는 데이터가 동기화 동작 동안에 다른 동기화 중단점들로 전파될 수 있다.

**대표도** - 도6C

630



(72) 발명자

**클락, 마이클, 레이**

미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내

**코스라비, 모에**

미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내

**노비크, 레프**

미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내

**펜닝, 조그-토머스**

미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내

## 특허청구의 범위

### 청구항 1

복수의 동기화 종단점들을 포함하는 동기화 커뮤니티에서 충돌 지식(conflict knowledge)을 전파하는 방법으로서,

제1 동기화 종단점에서, 제2 동기화 종단점의 지식 정보를 수신하는 단계 - 상기 지식 정보는 상기 제2 동기화 종단점이 알고 있는 상기 동기화 커뮤니티 내에서의 데이터에 대한 변경들을 나타냄 -,

상기 제2 동기화 종단점으로부터의 지식 정보를 상기 제1 동기화 종단점의 지식 정보와 비교하는 단계 - 상기 제1 동기화 종단점의 지식 정보는 상기 제1 동기화 종단점이 알고 있는 상기 동기화 커뮤니티 내에서의 데이터에 대한 변경들을 나타냄 -,

상기 제1 동기화 종단점에서, 상기 제2 동기화 종단점으로부터의 지식 정보와 상기 제1 동기화 종단점의 지식 정보의 비교에 기초하여 변경 정보를 열거하는 단계 - 상기 변경 정보는 제3 동기화 종단점과 상기 복수의 동기화 종단점들 중의 다른 동기화 종단점 간의 미해결된 동기화 데이터 충돌(unresolved synchronization data conflict)을 나타내는 데이터를 포함함 -, 및

상기 제1 동기화 종단점으로부터 상기 제2 동기화 종단점으로 상기 변경 정보를 전송하는 단계를 포함하는 동기화 커뮤니티에서 충돌 지식을 전파하는 방법.

### 청구항 2

제1항에 있어서,

상기 변경 정보를 전송하는 것과 동시에 또는 상기 변경 정보를 전송하는 단계 후에, 상기 제1 동기화 종단점의 지식 정보를 상기 제1 동기화 종단점으로부터 전송하는 단계,

상기 제1 동기화 종단점의 지식 정보의 상기 전송에 응답하여, 상기 제1 동기화 종단점에서, 상기 제2 동기화 종단점으로부터의 열거된 변경 정보를 수신하는 단계 - 상기 열거된 변경 정보는 제2의 미해결된 동기화 데이터 충돌을 나타내는 데이터를 포함함 -, 및

상기 제1 동기화 종단점이 상기 제2의 미해결된 동기화 데이터 충돌과 연관된 데이터 항목에 대해 행해진 변경에 대한 지식 - 이 지식은 상기 제2의 미해결된 동기화 데이터 충돌의 충돌 변경들에 대한 지식을 포함하는 연관된 '알고 있는지 여부(made-with-knowledge)'를 가짐 - 을 포함하는 경우에, 상기 제1 동기화 종단점에서 상기 제2의 미해결된 동기화 데이터 충돌을 해결하는 단계를 더 포함하는

동기화 커뮤니티에서 충돌 지식을 전파하는 방법.

### 청구항 3

제1항에 있어서,

상기 제2 동기화 종단점으로부터의 지식 정보는 지식 벡터를 포함하는

동기화 커뮤니티에서 충돌 지식을 전파하는 방법.

### 청구항 4

제1항에 있어서,

상기 변경 정보를 전송하는 것과 동시에 또는 상기 변경 정보를 전송하는 단계 후에, 상기 제1 동기화 종단점의 지식 정보를 상기 제1 동기화 종단점으로부터 전송하는 단계를 더 포함하는

동기화 커뮤니티에서 충돌 지식을 전파하는 방법.

### 청구항 5

제1항에 있어서,

상기 미해결된 동기화 데이터 충돌은 상기 제1 동기화 종단점과 제3 동기화 종단점 간의 동기화 중에 발생한 것

인

동기화 커뮤니티에서 충돌 지식을 전파하는 방법.

#### 청구항 6

제1항에 있어서,

상기 이해결된 동기화 데이터 충돌은 제3 동기화 중단점과 제4 동기화 중단점 간의 동기화 중에 발생한 것인

동기화 커뮤니티에서 충돌 지식을 전파하는 방법.

#### 청구항 7

적어도 하나의 프로세서에 대한 명령어들이 기록되어 있는 컴퓨터 판독가능저장 매체로서,

제1 동기화 중단점으로부터 제2 동기화 중단점으로 제1 동기화 요청을 전송하는 명령어들 - 상기 제1 동기화 요청은 상기 제1 동기화 중단점이 알고 있는 복수의 동기화 중단점들을 포함하는 동기화 커뮤니티 내에서의 데이터에 대한 변경들을 나타내는 지식 정보를 포함함 -,

상기 제1 동기화 요청의 전송에 응답하여, 상기 제1 동기화 중단점에서, 상기 제2 동기화 중단점은 알고 있으나 상기 제1 동기화 중단점은 알지 못하고 있는 상기 동기화 커뮤니티 내에서의 데이터에 대한 변경들을 나타내는 지식 정보를 포함하는 변경 정보를 제2 동기화 중단점으로부터 수신하는 명령어들,

상기 수신된 변경 정보에 적어도 부분적으로 기초하여, 상기 제1 동기화 중단점과 상기 제2 동기화 중단점 간에 동기화 데이터 충돌이 존재하는지를 판정하는 명령어들,

상기 동기화 커뮤니티 내의 제3 동기화 중단점으로 전파하기 위해 판정된 상기 동기화 데이터 충돌에 대한 지식을 저장하는 명령어들,

상기 제1 동기화 중단점에서, 복수의 동기화 중단점들 중 하나의 동기화 중단점으로부터 제2 동기화 요청을 수신하는 명령어들, 및

상기 제2 동기화 요청의 수신에 응답하여, 상기 제1 동기화 중단점과 상기 제2 동기화 중단점 간의 동기화 데이터 충돌을 나타내는 데이터를 포함하는 변경 정보를 상기 제1 동기화 중단점으로부터 상기 복수의 동기화 중단점들 중 상기 하나의 동기화 중단점으로 전송하는 명령어들을 포함하고,

상기 컴퓨터 판독가능 저장 매체는 처리 장치를 위한 메모리이거나 상기 처리 장치와 함께 사용하기 위한 저장 장치인

컴퓨터 판독가능 저장 매체.

#### 청구항 8

제7항에 있어서,

상기 복수의 동기화 중단점들 중 상기 하나의 동기화 중단점은 상기 제2 동기화 중단점 이외의 동기화 중단점인

컴퓨터 판독가능 저장 매체.

#### 청구항 9

제7항에 있어서,

상기 제2 동기화 요청은 상기 동기화 커뮤니티 내에서 상기 복수의 동기화 중단점들 중 상기 하나의 동기화 중단점이 알고 있는 동기화 데이터에 대한 변경들을 나타내는 지식 정보를 포함하는

컴퓨터 판독가능 저장 매체.

#### 청구항 10

제7항에 있어서,

상기 제1 동기화 중단점과 상기 제2 동기화 중단점 간의 동기화 데이터 충돌을 나타내는 데이터를 포함하는 상

기 변경 정보는 적어도 2개의 변경 ID들 및 대응하는 변경 데이터를 포함하는 충돌 변경 정보를 포함하는 컴퓨터 판독가능 저장 매체.

#### 청구항 11

제7항에 있어서,

상기 제1 동기화 요청에 포함된 상기 지식 정보는 지식 벡터를 포함하는

컴퓨터 판독가능 저장 매체.

#### 청구항 12

제7항에 있어서,

상기 변경 정보는 적어도 하나의 변경 ID 및 대응하는 변경 데이터를 포함하고,

상기 제1 동기화 중단점과 상기 제2 동기화 중단점 간에 동기화 데이터 충돌이 존재하는지를 판정하는 명령어들은,

상기 제2 동기화 중단점으로부터의 변경 정보가 데이터 항목에 대한 제1 변경을 포함하고, 상기 제1 동기화 중단점이 상기 데이터 항목에 대한 제2 변경을 알고 있으며, 상기 제2 동기화 중단점으로부터의 변경 정보에 포함되어 있으며 상기 데이터 항목에 대한 변경과 연관된 '알고 있는지 여부(made-with-knowledge)'가 상기 제2 동기화 중단점이 상기 데이터 항목에 대한 제2 변경을 알지 못하고 있음을 나타내고, 상기 데이터 항목에 대한 제2 변경과 연관되어 있는 상기 제1 동기화 중단점의 '알고 있는지 여부(made-with-knowledge)'가 상기 제1 동기화 중단점이 상기 데이터 항목에 대한 변경을 알지 못하고 있음을 나타내는 경우에, 상기 제1 동기화 중단점과 상기 제2 동기화 중단점 간에 동기화 데이터 충돌이 존재하는지를 판정하는 명령어들을 더 포함하는

컴퓨터 판독가능 저장 매체.

#### 청구항 13

제7항에 있어서,

상기 변경 정보는 적어도 하나의 변경 ID 및 대응하는 변경 데이터를 포함하고,

상기 적어도 하나의 변경 ID 각각은 버전 및 상기 동기화 커뮤니티 내의 특성의 동기화 중단점과 연관된 동기화 중단점 ID를 포함하는

컴퓨터 판독가능 저장 매체.

#### 청구항 14

제7항에 있어서,

상기 제2 동기화 중단점으로부터의 변경 정보에 대응하는 정보를 포함하는 예외 리스트를 저장하는 명령어들을 더 포함하는

컴퓨터 판독가능 저장 매체.

#### 청구항 15

제7항에 있어서,

상기 제2 동기화 중단점으로부터 수신된 변경 정보에 기초하여 상기 제1 동기화 중단점의 지식 정보를 갱신하는 명령어들을 더 포함하는

컴퓨터 판독가능 저장 매체.

#### 청구항 16

복수의 동기화 중단점들을 포함하는 동기화 커뮤니티 내에 제1 동기화 중단점을 구현하는 처리 장치로서,

적어도 하나의 프로세서,  
통신 인터페이스, 및  
상기 적어도 하나의 프로세서에 대한 명령어들을 포함하는 메모리를 포함하며,  
상기 적어도 하나의 프로세서와 상기 메모리는 상기 통신 인터페이스를 통해 접속되어 있고,  
상기 적어도 하나의 프로세서에 대한 명령어들은,  
상기 제1 동기화 종단점이 알고 있는 상기 동기화 커뮤니티 내에서의 데이터에 대한 변경들을 나타내는 상기 제1 동기화 종단점의 지식 정보를 유지하는 명령어들,  
상기 제1 동기화 종단점의 지식 정보를 제2 동기화 종단점으로부터 수신된 지식 정보와 비교하는 명령어들,  
상기 제1 동기화 종단점의 지식 정보와 상기 제2 동기화 종단점으로부터 수신된 지식 정보의 비교에 기초하여, 동기화 데이터 충돌이 존재하는지를 판정하는 명령어들, 및  
제3 동기화 종단점으로 전파하기 위해 판정된 상기 동기화 데이터 충돌에 대한 지식을 저장하는 명령어들을 포함하는  
동기화 커뮤니티 내에 제1 동기화 종단점을 구현하는 처리 장치.

#### 청구항 17

제16항에 있어서,  
상기 제1 동기화 종단점의 지식 정보는 제1 지식 벡터를 포함하고,  
상기 제2 동기화 종단점의 지식 정보는 제2 지식 벡터를 포함하는  
동기화 커뮤니티 내에 제1 동기화 종단점을 구현하는 처리 장치.

#### 청구항 18

제16항에 있어서,  
상기 제1 동기화 종단점의 지식 정보 및 상기 제2 동기화 종단점의 지식 정보는 각자의 상기 동기화 종단점들이 알고 있는 상기 동기화 커뮤니티의 데이터 변경 정보, 및 상기 동기화 데이터 변경 정보의 각각의 항목에 대응하는 '알고 있는지 여부(made-with-knowledge)' 정보를 포함하는  
동기화 커뮤니티 내에 제1 동기화 종단점을 구현하는 처리 장치.

#### 청구항 19

제16항에 있어서,  
상기 제1 동기화 종단점의 지식 정보 및 상기 제2 동기화 종단점의 지식 정보는 각자의 상기 동기화 종단점들이 알고 있는 상기 동기화 커뮤니티의 데이터 변경 정보, 및 상기 동기화 데이터 변경 정보의 각각의 항목에 대응하는 '알고 있는지 여부(made-with-knowledge)' 정보를 포함하고,  
상기 메모리는,  
상기 제2 동기화 종단점으로부터의 동기화 데이터 변경 정보가 데이터 항목에 대한 제1 변경을 포함하고, 상기 제1 동기화 종단점이 상기 데이터 항목에 대한 제2 변경을 알고 있으며, 상기 제2 동기화 종단점으로부터의 동기화 데이터 변경 정보에 포함되어 있으며 상기 데이터 항목에 대한 제1 변경과 연관된 '알고 있는지 여부(made-with-knowledge)'가 상기 제2 동기화 종단점이 상기 데이터 항목에 대한 제2 변경을 알지 못하고 있음을 나타내고, 상기 데이터 항목에 대한 제2 변경과 연관되어 있는 상기 제1 동기화 종단점의 '알고 있는지 여부(made-with-knowledge)'가 상기 제1 동기화 종단점이 상기 데이터 항목에 대한 변경을 알지 못하고 있음을 나타낼 때, 상기 제1 동기화 종단점과 상기 제2 동기화 종단점 간에 동기화 데이터 충돌이 존재하는지를 판정하는 명령어들을 더 포함하는  
동기화 커뮤니티 내에 제1 동기화 종단점을 구현하는 처리 장치.

## 청구항 20

제16항에 있어서,

상기 메모리는 상기 제1 동기화 종단점의 지식 정보와 상기 제3 동기화 종단점의 지식 정보의 비교에 기초하여 변경 정보를 열거하는 명령어들을 더 포함하며,

상기 변경 정보는 상기 동기화 데이터 충돌을 나타내는 정보를 포함하는

동기화 커뮤니티 내에 제1 동기화 종단점을 구현하는 처리 장치.

## 명세서

### 기술분야

[0001] 본 발명은 충돌 지식의 전파 방법에 관한 것이다.

### 배경기술

[0002] 동기화 종단점(synchronization endpoint)은 처리 장치 상에 구현될 수 있고, 컴퓨터, 노트북 컴퓨터, PDA(personal digital assistant), 셀룰러 전화 또는 기타 무선 장치, 하나 이상의 파일을 갖는 저장 장치 상의 폴더, 메모리 내의 일군의 기억 장소, 서버 컴퓨터, 온라인 서비스, 데이터가 전자적으로 저장될 수 있거나 데이터가 전자적으로 검색될 수 있는 임의의 기타 장치 또는 컨테이너, 또는 이들의 임의의 조합(이들로 제한되지 않음)을 포함할 수 있는 다양한 데이터 저장소 또는 장치를 나타낼 수 있다. 동기화 커뮤니티(sync community)는 서로 동기화할 수 있는 일군의 동기화 종단점일 수 있다. 양방향 멀티-마스터 동기화 토폴로지(two-way multi-master synchronization topology)에서, 동기화 종단점 쌍의 제1 동기화 종단점이 그 동기화 종단점 쌍의 제2 동기화 종단점과의 동기화를 요청할 수 있고 또 제2 동기화 종단점이 제1 동기화 종단점과의 동기화를 요청할 수 있는 방식으로 동기화 종단점 쌍이 서로 데이터를 동기화시킬 수 있다. 양방향 멀티-마스터 동기화 토폴로지에서도, 서로 다른 종단점들에서 특정의 데이터 항목에 대해 동시적인 또는 거의 동시적인 수정이 행해질 수 있으며, 그에 따라 그 종단점들이 동기화를 시도할 때 충돌이 발생할 수 있다.

[0003] 기존의 동기화 애플리케이션은 충돌을 해결하기 위해 다수의 기법들을 사용한다. 어떤 동기화 애플리케이션들은 동기화 시에 사용자 인터페이스(UI)를 디스플레이하고 사용자가 충돌 해결을 선택하게 함으로써 충돌을 해결한다. 이러한 접근방법의 단점은 사용자가 UI에 응답할 수 있도록 동기화 프로세스 동안에 사용자가 자리에 있어야만 하거나 동기화 프로세스를 완료하지 못할 위험이 있다는 것이다. 다른 동기화 애플리케이션들은 충돌 해결을 지연시키고 사용자가 나중에 UI를 통해 충돌 해결을 할 수 있게 해준다. 어떤 상황들에서는, 지연된 충돌 해결이 결코 해결될 수 없는데, 그 이유는 그 충돌이 충돌을 해결하기 위한 기능을 갖지 않는 종단점에서 검출될지도 모르기 때문이다. 다른 동기화 애플리케이션은 자동 충돌 해결 정책을 구현할 수 있다. 예를 들어, 동기화 애플리케이션은 더 늦은 기록 시간을 갖는 정정을 충돌의 승리자로 항상 선택함으로써 충돌을 해결할 수 있다. 이러한 동기화 애플리케이션에서, 폐기된 정정은 실행 취소 동작을 위해 저장될 수 있으며, 이 실행 취소는, 수행될 때, 폐기된 정정을 충돌의 승리자로 선언되게 할 수 있으며 그에 따라 데이터를 정정할 수 있다. 그렇지만, 이러한 동기화 애플리케이션의 단점은 실행 취소 동작이 항목에 대해 수행된 하나 이상의 행위들을 되돌리는 것이 아니라 항목에 대한 새로운 정정으로서 기능한다는 것이다.

### 발명의 상세한 설명

[0004] 이 요약은 이하에서 상세한 설명에 더 기술되는 일련의 개념들을 간략화된 형태로 소개하기 위해 제공된 것이다. 이 요약은 청구된 발명 대상의 주요 특징들 또는 필수적인 특징들을 확인하기 위한 것이 아니며, 청구된 발명 대상의 범위를 제한하는 데 사용되기 위한 것도 아니다.

[0005] 본 발명에 따른 실시예들에서, 동기화 커뮤니티 내의 동기화 종단점들은 동기화 종단점들이 일관성있는 최신 데이터를 포함할 수 있도록 서로 동기화할 수 있다. 그렇지만, 2개의 동기화 종단점은 각각의 동기화 종단점이 각자의 상대방 동기화 종단점에 의해 행해진 변경을 알지 못하고 있을 때 동일한 데이터 항목을 변경할지도 모른다. 이러한 변경이 동일한 데이터 항목에 행해질 때, 동기화 데이터 충돌이 존재할 수 있다. 동기화 데이터 충돌의 해결이 지연될 수 있고, 다른 동기화 종단점들과 동기화할 때 동기화 데이터 충돌에 대한 지식이 그 다른 동기화 종단점들로 전파될 수 있다.

## 실시예

- [0015] 상기한 이점들 및 특징들과 기타의 이점들 및 특징들이 달성될 수 있는 방식을 설명하기 위해, 첨부된 도면들에 예시되어 있는 본 발명의 구체적인 실시예들을 참조하여 이하에 보다 상세한 설명이 기술된다. 이들 도면이 단지 통상적인 실시예들을 나타낸 것이며 따라서 본 발명의 범위를 제한하는 것으로 생각되어서는 안된다는 것을 이해하면서, 첨부 도면들을 사용하여 여러 구현예들을 더 구체적이고 상세하게 기술하고 설명할 것이다.
- [0016] 이하에서 실시예들에 대해 상세히 설명한다. 특정의 구현예들이 기술되어 있지만, 이것이 단지 예시를 위한 것임을 잘 알 것이다. 당업자라면 본 발명의 정신 및 범위를 벗어나지 않고 다른 컴포넌트들 및 구성들이 사용될 수 있다는 것을 잘 알 것이다.
- [0017] 예시적인 운영 환경
- [0018] 본 발명에 따른 실시예들은 동기화 커뮤니티(sync community)의 동기화 종단점들(synchronization endpoints) 간의 데이터 동기화에 관한 것이다. 동기화 종단점은 처리 장치 상에 구현될 수 있고, 컴퓨터, 노트북 컴퓨터, PDA(personal digital assistant), 셀룰러 전화 또는 기타 무선 장치, 하나 이상의 파일을 갖는 저장 장치 상의 폴더, 메모리 내의 일군의 기억 장소, 서버 컴퓨터, 온라인 서비스, 데이터가 전자적으로 저장될 수 있거나 데이터가 전자적으로 검색될 수 있는 임의의 기타 장치 또는 컨테이너, 또는 이들의 임의의 조합(이들로 제한되지 않음)을 포함할 수 있는 다양한 데이터 저장소 또는 장치일 수 있다. 어떤 실시예들에서, 동기화 종단점을 구현하는 장치는 다수의 동기화 종단점들을 포함할 수 있다. 2개의 동기화 종단점들 간에 행해지는 동기화 동작의 예로는 2개의 처리 장치, 하나의 저장 장치 또는 2개의 서로 다른 저장 장치 상의 2개의 폴더, 또는 하나의 메모리 또는 2개의 서로 다른 메모리 내의 2 그룹의 메모리 장소들의 동기화가 있을 수 있지만, 이들로 제한되지 않는다.
- [0019] 도 1은 동기화 커뮤니티 내의 하나 이상의 동기화 종단점을 구현하는 데 사용될 수 있는 처리 장치(100)의 기능 블록도이다. 처리 장치(100)는 버스(110), 프로세서(120), 메모리(130), 판독 전용 메모리(ROM)(140), 저장 장치(150), 입력 장치(160), 출력 장치(170), 및 통신 인터페이스(180)를 포함할 수 있다. 버스(110)는 처리 장치(100)의 컴포넌트들 간의 통신을 가능하게 해주는 통신 인터페이스일 수 있다.
- [0020] 프로세서(120)는 명령어들을 해석하여 실행하는 적어도 하나의 종래의 프로세서 또는 마이크로프로세서를 포함할 수 있다. 메모리(130)는 프로세서(120)에서 실행하기 위한 명령어들 및 정보를 저장하는 랜덤 액세스 메모리(RAM) 또는 기타 유형의 동적 저장 장치일 수 있다. 메모리(130)는 또한 프로세서(120)에서 명령어들을 실행하는 동안에 사용되는 임시 변수 또는 기타 중간 정보를 저장할 수 있다. ROM(140)은 프로세서(120)에 대한 정적 정보 및 명령어들을 저장하는 종래의 ROM 장치 또는 기타 유형의 정적 저장 장치를 포함할 수 있다. 저장 장치(150)는 데이터 및/또는 명령어들을 저장하는 임의의 종류의 유형 매체(tangible media)를 포함할 수 있다. 어떤 실시예들에서, 저장 장치(150)는 하나 이상의 폴더를 포함할 수 있으며, 각각의 폴더는 하나 이상의 파일을 포함할 수 있다.
- [0021] 입력 장치(160)는 사용자가 처리 장치(100)에 정보를 입력할 수 있게 해주는 하나 이상의 종래의 메카니즘(예를 들어, 키보드, 마우스, 또는 기타 입력 장치 등)을 포함할 수 있다. 출력 장치(170)는 사용자에게 정보를 출력하는 하나 이상의 종래의 메카니즘(예를 들어, 디스플레이, 프린터, 또는 기타 출력 장치 등)을 포함할 수 있다. 통신 인터페이스(180)는 처리 장치(100)가 다른 장치들 또는 네트워크들과 통신할 수 있게 해주는 임의의 송수신기와 같은 메카니즘을 포함할 수 있다.
- [0022] 처리 장치(100)는 프로세서(120)가 기계 판독가능 매체(예를 들어, 메모리(130) 또는 기타의 유형 매체 등)에 포함된 명령어들의 시퀀스를 실행한 것에 응답하여 이러한 기능들을 수행할 수 있다. 이러한 명령어들은 다른 기계 판독가능한 유형 매체(저장 장치(150) 등)로부터 또는 통신 인터페이스(180)를 통해 별도의 장치로부터 메모리(130) 내로 판독될 수 있다.
- [0023] 도 2는 동기화 종단점들의 동기화 커뮤니티에 대한 예시적인 운영 환경(200)을 나타낸 것이다. 예시적인 운영 환경(200)은 네트워크(202), 동기화 종단점(204), 및 동기화 종단점(206)을 포함할 수 있다.
- [0024] 네트워크(202)는 유선 또는 무선 네트워크일 수 있고, 유선 또는 무선 수단을 통해 접속된 다수의 장치들을 포함할 수 있다. 네트워크(202)는 단 하나의 네트워크 또는 다수의 서로 다른 네트워크들(이들 중 일부는 상이한 종류의 네트워크일 수 있음)을 포함할 수 있다.
- [0025] 동기화 종단점(204) 및 동기화 종단점(206)은 2개의 상이한 처리 장치 상(이 예시적인 실시예에서, 예를 들어,

처리 장치(100) 등에 구현될 수 있다. 예시적인 실시예(200)에서, 동기화 중단점(204) 및 동기화 중단점(206)은 네트워크(202)를 통해 서로 통신을 할 수 있다.

[0026] 운영 환경(200)은 예시적인 것이다. 본 발명에 따른 다른 실시예들은 다른 운영 환경들에서 구현될 수 있다. 예를 들어, 어떤 운영 환경들(예를 들어, 동기화 중단점들이 단일의 장치 내에 포함되어 있을 수 있는 운영 환경 등)에서는 네트워크(202)가 포함되어 있지 않을 수 있다.

[0027] 동기화

[0028] 본 발명에 따른 실시예들에서, 동기화 중단점들은 일관성있는 최신의 데이터를 포함할 수 있도록 서로 동기화를 할 수 있다. 하나의 동기화 중단점이 특정의 데이터 항목에 대한 변경을 포함하고, 다른 동기화 중단점이 그 특정의 데이터 항목에 대한 다른 변경을 포함하며, 어느 동기화 중단점도 각자의 상대방 동기화 중단점에 의해 그 특정의 데이터 항목에 행해진 변경을 알지 못한 경우, 동기화 중단점들의 동기화 동안에 충돌이 일어날 수 있다. 예를 들어, 동기화 중단점 1은 주소록 데이터베이스에서 John Smith의 전화 번호를 갱신할 수 있고, 동기화 중단점 2는 주소록 데이터베이스에서 John Smith의 전화 번호를 삭제할 수 있다. 동기화 중단점 1 및 동기화 중단점 2 둘다가 각자의 상대방 동기화 중단점에 의해 행해진 변경을 알지 못하고 상기한 변경을 한 경우, 이들 동기화 중단점이 서로 동기화를 할 때, 충돌이 일어날 수 있다. 본 발명에 따른 다양한 실시예들에서, 동기화 중단점이 충돌을 해결할 수 없는 경우, 충돌 해결이 지연될 수 있고, 충돌에 대한 지식이 동기화 동작 동안에 다른 동기화 중단점들로 전파될 수 있다.

[0029] 동기화는 통상적으로 동기화 커뮤니티를 형성하는 한 그룹의 참여 중인 동기화 중단점들 간에 행해진다. 동기화 커뮤니티의 총 멤버수(total membership)가 임의의 주어진 때에 임의의 주어진 동기화 중단점에 알려져 있지 않을 수 있다. 게다가, 동기화 커뮤니티의 토폴로지가 임의의 주어진 때에 임의의 주어진 동기화 중단점에 알려져 있지 않을 수 있다. 일 실시예에서, 동기화 커뮤니티 내의 각각의 동기화 중단점은, 예를 들어, GUID(global unique identifier)와 같은 ID, 또는 동기화 커뮤니티 내에서 고유할 수 있는 기타 ID를 가질 수 있다.

[0030] 각각의 동기화 중단점은 동기화 중단점들 간의 효율적이고 개선된 동기화를 용이하게 해주는 "지식"을 유지할 수 있다. 일 실시예에서, 지식은 주어진 동기화 중단점이 알고 있는 변경을 표현하는 메타데이터일 수 있다. 지식은 변경 ID의 벡터로서 표현될 수 있으며, 여기서 각각의 변경 ID는 동기화 중단점 ID와 가장 높은 버전(동기화 중단점 ID, 가장 높은 버전)을 나타낼 수 있다. 특정의 지식 벡터에서의 변경 ID의 수는 동기화 중단점이 동기화 커뮤니티에 추가되거나 그로부터 제거될 때 변경될 수 있다. 지식 벡터가 다르게 표현될 수도 있지만, 특정의 동기화 중단점이 알고 있는 변경을 간결하게 표현하는 것이 유익하다. 특정의 지식이 동기화 커뮤니티 내의 각각의 동기화 중단점에 대한 변경 ID를 특별히 포함하거나 포함하지 않을 수 있다. 동기화 중단점은 다른 동기화 중단점들이 이미 알고 있는 것을 추적하는 것으로부터 해방될 수 있는데, 그 이유는 이러한 정보가 동기화 중단점의 지식에 의해 효과적으로 표현되기 때문이다.

[0031] 동기화 커뮤니티 내의 동기화 중단점들은 그 자신의 지식을 자기와 동기화하는 동기화 중단점에 제공함으로써 동기화할 수 있다. 동기화 중인 동기화 중단점들 간에 전송될 수 있는 지식을 표현하는 데이터의 양을 줄이기 위해, 지식이 앞서 기술된 바와 같은 지식 벡터로서 표현될 수 있다. 따라서, 동기화 중단점들 간에 전송되는 지식은 모든 변경 ID를 포함할 필요가 없고 다수의 변경 ID를 표현하는 벡터의 형태로 되어 있을 수 있다. 예를 들어, 동기화 중단점이 제1 변경에서부터 제10 변경까지의 동기화 중단점 A에 의해 행해진 모든 변경 및 제1 변경에서부터 제5 변경까지의 동기화 중단점 B에 의해 행해진 모든 변경을 알고 있는 경우, 동기화 중단점은 변경 ID(A1 내지 A10)에 대응하는 모든 변경 및 변경 ID(B1 내지 B5)에 대응하는 모든 변경을 알고 있다는 것을 나타내는 지식 벡터(A10B5)를 전송할 수 있다. 지식이 지식 벡터로 표현될 수 있지만, 다른 실시예들은 지식의 다른 표현들도 생각하고 있다. 예를 들어, 어떤 실시예들은 (1) 변경을 지식 표현에 추가하고, (2) 변경이 지식 표현에 포함되어 있는지를 검사하며, (3) 2개의 지식 표현을 서로 병합할 수 있게 되어 있는 임의의 지식 표현을 사용하여 지식을 표현한다.

[0032] 도 3은 예시된 토폴로지를 갖는 동기화 커뮤니티(300)의 일례를 나타낸 것이다. 동기화 커뮤니티(300)는 다수의 동기화 중단점들을 포함하며, 본 발명에 따른 실시예들을 구현하는 환경의 일례이다. 동기화 커뮤니티(300) 내의 동기화 중단점들은, 예를 들어, 처리 장치(100) 등의 장치 상에 구현될 수 있고, 컴퓨터, 노트북 컴퓨터, PDA, 셀룰러 전화 또는 기타 무선 장치, 하나 이상의 파일을 갖는 저장 장치 상의 폴더, 메모리 내의 일군의 기억 장소, 서버 컴퓨터, 온라인 서비스, 데이터가 전자적으로 저장될 수 있거나 데이터가 전자적으로 검색될 수 있는 임의의 다른 장치 또는 컨테이너, 또는 이들의 임의의 조합(이들로 제한되지 않음)을 포함할 수 있는 다양

한 데이터 저장소 또는 장치일 수 있다.

[0033] 도 3은 예시적인 동기화 커뮤니티(300)를 나타낸 것이다. 도 3에서, 동기화 종단점 A(302)는 통신 링크(306)를 통해 동기화 종단점 B(304)에 전자적으로 결합되어 있을 수 있다. 동기화 종단점 A(302)는 통신 링크(308)를 통해 동기화 종단점 C(310)에 접속되어 있을 수 있다. 동기화 종단점 C(310)는 통신 링크(312)를 통해 동기화 종단점 B(304)에 접속되어 있을 수 있다. 동기화 종단점 C(310)는 또한 통신 링크(316)를 통해 동기화 종단점 D(314)에 접속되어 있을 수 있다. 동기화 커뮤니티(300)에서, 모든 동기화 종단점들이 통신 링크를 통해 직접 접속되어 있는 것은 아니지만, 동기화 종단점들 중의 임의의 동기화 종단점에서의 변경이 동기화 커뮤니티(300) 내의 나머지 동기화 종단점들 중 임의의 동기화 종단점과 동기화될 수 있다.

[0034] 예를 들어, 동기화 종단점 A(302)가 동기화 종단점 D(314)와 동기화되기 위해, 동기화 종단점 A(302)와 동기화 종단점 C(310)는 통신 링크(308)를 통해 동기화될 수 있다. 따라서, 동기화 종단점 C(310)는 동기화 종단점 A(302)에서 행해진 변경을 포함할 수 있다. 이어서, 동기화 종단점 C(310) 및 동기화 종단점 D(314)가 통신 링크(316)를 통해 동기화될 수 있고, 따라서 동기화 종단점 D(314)가 동기화 종단점 A(302)로부터의 변경을 포함할 수 있다. 이와 같이, 어떤 종류의 직접 링크도 없이 동기화 종단점 A(302)가 동기화 종단점 D(314)와 동기화될 수 있다. 실제로, 동기화 종단점 A(302)와 동기화 종단점 D(314)는 동기화 커뮤니티(300) 내에 상대방이 존재하는지조차도 모를 수 있다. 예시된 통신 링크들은 유선 링크 및/또는 무선 링크일 수 있다.

[0035] 이제 도 4를 참조하면, 본 발명에 따른 일 실시예는 동기화 종단점에서 변경들이 어떻게 관리될 수 있는지를 나타낸 것이다. 도 4는 동기화 종단점 A(400)의 시간별 진행을 나타낸 것이다. 동기화 종단점 A(400)는 지식(402)(이 경우에  $K_A$ 로 표시됨) 및 변경들(404)(이 경우에,  $\Delta_A$ 로 표시됨)을 포함할 수 있다. 변경들(404) 내의 각각의 변경은 항목의 최신의 데이터 내용일 수 있다. 변경은 동기화 종단점에 추가된 새로운 항목(항목 자체가 변경되지는 않았음), 항목의 삭제, 기타 등등일 수 있다. 변경들(404) 중의 각각의 변경은, 일 실시예에서, 변경 ID일 수 있는 버전과 연관되어 있을 수 있다. 시간 (1)에서, 동기화 종단점 A(400)는 안정 상태에 있다. 시간 (2)에서, 사용자가 변경(X로 표시됨)을 동기화 종단점 A(400)에 입력한다. 도 4는 변경(X)이 변경들(404)의 멤버로서 추가되는 것을 나타내고 있다. 시간 (3)에서, 지식(402)은 변경(X)과 연관되어 있는 변경 ID(ChangeID(X))를 포함하도록 갱신될 수 있고, 변경(X)이 변경들(404)에 추가된 것을 알려줄 수 있다. 이 실시예는 동기화 종단점에 대한 변경이 특정의 변경 ID와 연관될 수 있는 한 방법을 나타낸 것이다. 지식(404)은 지식 벡터일 수 있고, 동기화 종단점 A(400)가 알고 있는 변경을 나타낼 수 있다. 일 실시예에서, 항목 또는 객체에 대한 버전, 즉 변경 ID가 데이터베이스에 유지될 수 있고, 이 버전은 어느 것이 동기화되어야 하는지를 식별하는 데 사용될 수 있다. 다른 대안으로서, 변경들의 로그도 역시 유지될 수 있다.

[0036] 도 5는 동기화 동안에 지식을 사용하여 변경들을 열거하는 것을 나타낸 것이다. 도 5는 2개의 동기화 종단점, 즉 동기화 종단점 A(502) 및 동기화 종단점 B(504)를 나타내고 있다. 동기화 종단점 A(502)는 일련의 변경들(506)(이 예에서,  $\Delta_A$ 로 표시됨)을 포함할 수 있다. 동기화 종단점 A(502)는 또한 지식(508)(이 예에서,  $K_A$ 로 표시됨)을 포함할 수 있다. 지식(508)은 상기한 것들과 같은 변경 ID들의 리스트를 포함할 수 있다. 이와 유사하게, 동기화 종단점 B(504)는 일련의 변경들(510)을 포함할 수 있으며, 각각의 변경은 변경 ID인 버전과 연관되어 있을 수 있다. 동기화를 시작하기 위해, 시간 (1)에서, 동기화 종단점 A(502)는 지식(508)을 포함할 수 있는 동기화 요청(synch request)을 동기화 종단점 B(504)로 전송할 수 있다. 동기화 종단점 B(504)는, 지식(508)을 일련의 변경들(510) 내의 각각의 변경과 연관된 버전들과 비교함으로써, 동기화 종단점 A(502)가 동기화 종단점 B의 변경들(510) 중 어느 것을 그의 변경들(506) 내에 이미 가지고 있는지 및 동기화 종단점 A(502)가 알고 있는 변경들에 관해 결정을 할 수 있다. 다른 대안으로서, 동기화 종단점 B(504)는 지식(508)을 각각의 항목의 버전과 비교할 수 있다. 따라서, 동기화 종단점 B(504)는 동기화 종단점 A(502)의 지식(508)에 포함되지 않은 버전들과 연관된 동기화 종단점 B의 변경들(510)의 그 부분만(변경들(514)로 표시됨)을 열거하여 시간 (2)에서 동기화 종단점 A(502)로 전송할 수 있다. 예를 들어, 동기화 종단점(502)의 지식 벡터가 A3B12이고 동기화 종단점 B(504)가 변경 ID(B14, B14)인 버전들과 연관된 최신의 변경들을 갖는 경우, 동기화 종단점 A(502)로 전송되는 변경들은 변경 ID(B13, B14)와 연관된 것들을 포함하게 된다. 일 실시예에서, B13 및 B14가 동일한 항목에 대해 행해진 경우 B14만이 전송될 수 있다.

[0037] 그에 부가하여, 동기화 종단점 B(504)는 또한 동기화 종단점 B의 지식(512)을 동기화 종단점 A(502)로 전송할 수 있다. 동기화 종단점 B(504)가 동기화 종단점 A(502)에는 아직 없고 동기화 종단점 B(504)에는 있는 변경들(510) 모두를 동기화 종단점 A(502)로 전송했기 때문에, 그 변경들(510)이 동기화 종단점 B(504)에 의해 전송된 변경들에 의해 대체되지 않은 한, 원래 동기화 종단점 B(504)에 있었던 변경들(510)에 부가하여, 동기화 종단점

A(502)는 이제 원래 동기화 중단점 A(502)에 있었던 변경들(506) 모두를 갖는다. 동기화 중단점 A(502)는 또한 동기화 중단점 B(504)가 알고 있었던 변경들 모두에 관한 정보를 갖는다. 따라서, 동기화 중단점 A(502)는 변경들(502)의 추가를 반영하기 위해 그의 지식(508)을 갱신할 수 있다. 이것은, 도 5의 시간 (3)에 나타난 바와 같이, 단순히 동기화 중단점 A의 지식(508)을 동기화 중단점 B의 지식(512)에 추가하고 그 값을 동기화 중단점 A의 지식(508)으로서 정의함으로써 행해진다.

[0038]

이에 따라, 효율적인 동기화가 수행될 수 있으며, 이 경우 필요한 변경들만이 동기화되고 개개의 동기화 중단점들은 그 특정의 동기화 중단점 내에 존재하는 변경들 및 그 특정의 동기화 중단점이 알고 있는 이전의 변경들에 관한 정보만 유지하고 있으면 된다. 이 예가 동기화 중단점 B(504)에서의 변경들 모두를 동기화 중단점 A(502)와 완전히 동기화시키는 것을 나타내고 있지만, 변경들 중 일부분만이 동기화되는 경우들이 존재할 수 있다. 따라서, 동기화되는 변경들에 대응하는 변경 ID만이 갱신을 수신하는 동기화 중단점의 지식에 추가될 수 있다.

[0039]

변경들을 열거하는 것에 부가하여, 동기화 중단점의 지식은 충돌 검출에도 사용될 수 있다. 이제 도 6A 및 도 6B를 참조하면, 본 발명의 일 실시예는 충돌 검출이 어떻게 달성될 수 있는지를 나타낸 것이다. 도 6A는 2개의 동기화 중단점이 통신 및 동기화를 위해 통신 링크(무선 및/또는 유선)에 의해 접속되어 있는 것을 나타낸 것이다. 동기화 중단점 A(602)는 지식(608) 및 일련의 변경들(606)을 포함할 수 있다. 도 5의 예에서와 같이, 지식(608)은 변경들(606)과 연관되고 또 이전의 변경들과 연관되어 있는 변경 ID들의 모음을 포함할 수 있다. 동기화 중단점 A(602)는 또한, 이 예의 목적상, 동기화 중단점 A(602)에서 행해진 항목에 대한 변경을 포함할 수 있다. 이 변경은 도 6A에 X로 나타내어져 있고, X는 변경들(606)의 멤버이다. 이와 유사하게, 동기화 중단점 B(604)는 지식(612), 변경들(610)의 모음, 및 변경들(610)의 멤버인 항목에 대한 변경(Y로 표시됨)을 포함할 수 있다.

[0040]

일례로서, 도 6B의 동작(620)에서, 동기화 중단점 A(602)는 변경(X로 표시됨)을 동기화 중단점 B(604)로 전송할 수 있다. 2개의 다른 값, 즉 변경(X)과 연관된 변경 ID(ChangeID(X)로 표시됨)와, 동기화 중단점 A(602)에 변경(X)이 행해졌을 때 동기화 중단점 A(602)에 존재했던 지식인 '알고 있는지 여부'(made-with-knowledge) 값( $K_A(X)$ 로 표시됨)이 변경(X)과 연관되고 변경(X)과 함께 전송될 수 있다(동작 620). 다른 대안으로서, 본 발명에 따른 어떤 실시예들에서, '알고 있는지 여부'(made-with-knowledge)는 변경이 전송될 때 동기화 중단점에 존재했던 지식일 수 있다. 동기화 중단점 A(602)의 최신의 지식(608)도 역시 동기화 중단점 B(604)로 전송될 수 있다. 동기화 중단점 B(604)는 이어서 변경(X)에 의해 변경된 항목( $I_X$ )을 변경(Y)에 의해 변경된 항목( $I_Y$ )과 비교할 수 있다(동작 622). 변경(X) 및 변경(Y)가 서로 다른 항목에 대응하는 경우, 충돌이 없다(동작 624).

[0041]

변경들이 동일한 항목의 서로 다른 버전들에 대한 것인 경우, 추가적인 분석은 충돌이 발생한 것으로 판정할 수 있다. 동기화 중단점 B(604)는 이어서 동기화 중단점 B(604)에서 변경(Y)이 행해졌을 때 변경(X)을 동기화 중단점 B(604)가 알고 있는지를 알아보기 위해 검사를 할 수 있다(동작 626). 변경(Y)는 변경 ID(ChangeID(Y)) 및 그와 연관된 '알고 있는지 여부'(made-with-knowledge) 값( $K_B(Y)$ )을 가질 수 있다. ChangeID(X)가 변경(Y)의 '알고 있는지 여부'(made-with-knowledge)( $K_B(Y)$ )의 멤버인 경우, 충돌이 없다. 환언하면, 동기화 중단점 A(602)에서 변경(X)이 행해졌다는 것을 알고서 동기화 중단점 B(604)에서 변경(Y)이 행해졌다. 따라서, 변경(Y)은 이제 동기화 중단점 A(602) 및 동기화 중단점 B(604)에 대한 가장 최근의 유효한 데이터를 나타낸다. 도 6B에 예시된 예에 도시되어 있지는 않지만, 나중에 변경(Y)이 동기화 중단점 A(602)로 전송되고, 변경(X) 및 변경(Y)과 연관된 항목이 도 5에 기술된 방식으로 동기화 중단점 A(602)에서 변경(Y)으로 갱신될 가능성이 있다.

[0042]

변경(X) 및 변경(Y)이 동일한 항목에 대한 것이고 ChangeID(X)가  $K_B(Y)$ 에 나타나지 않는 경우, 변경(X)이 행해졌을 때 동기화 중단점 A(602)가 변경(Y)을 알고 있었는지를 알아보기 위해 검사가 행해진다(동작 628). 이것은 통상적으로 ChangeID(Y)로 나타난 변경(Y)에 대한 변경 열거(change enumeration)가 변경(X)이 행해졌을 때 동기화 중단점 A의 지식(608)( $K_A(X)$ )에 포함되어 있는지를 알아보기 위해 검사를 함으로써 행해진다. ChangeID(Y)가  $K_A(X)$ 의 멤버인 경우, 변경(X)은 변경(Y)의 '알고 있는지 여부'(made-with-knowledge)이었고, 충돌이 없다(동작 624). 그렇지 않은 경우, 충돌이 존재하는 것으로 판정된다(동작 630). 이 예에서, 변경(X)이 특정의 항목에 대한 가장 최근의 유효한 변경이다. 따라서, 동기화 중단점 B(604)는 도 5에 기술된 방식으로 변경(X)을 사용하여 갱신될 가능성이 있다.

[0043]

변경(X) 및 변경(Y)이 동일한 항목에 대한 것이고 ChangeID(Y)가  $K_A(X)$ 에 나타나지 않으며 ChangeID(X)가  $K_B(Y)$ 에 나타나지 않는 경우, 정말로 충돌이 존재한다. 환언하면, 변경(X) 및 변경(Y)이 서로 독립적으로 행해졌

다. 이 경우에, 충돌이 검출되며, 어느 변경(X 또는 Y)이 가장 최근의 유효한 변경인지를 판정하기 위해 다양한 충돌 해결 규칙들이 적용될 수 있다. 이러한 규칙들은 어느 변경이 가장 최근에 행해졌는지를 판정하기 위해 타임 스탬프들을 검사하는 것, 어떤 종류의 동기화 중단점(서버에 저장된 것 등)에 항상 유리하게 충돌을 해결하는 것, 및/또는 임의의 다른 적당한 충돌 해결을 포함할 수 있다. 다른 대안으로서, 한 형태의 충돌 해결에서, 충돌하는 변경들을 갖는 항목은 충돌하는 변경들이 병합되어 새로운 변경을 형성하도록 갱신될 수 있다.

[0044]

도 6C는, 도 6B의 동작(630) 동안에 충돌이 검출될 때, 동기화 중단점(예를 들어, 동기화 중단점 B(604) 등)에서 일어날 수 있는 예시적인 처리를 나타낸 플로우차트이다. 충돌이 존재하는 것으로 판정한 후에, 동기화 중단점 B(604)는 판정된 충돌을 해결하기 위한 충돌 해결 정책 또는 규칙이 존재하는지를 판정할 수 있다(동작 632). 충돌 해결 정책 또는 규칙이 존재하는 것으로 판정된 경우, 동기화 중단점 B(604)는 충돌 해결 정책 또는 규칙에 따라 충돌을 해결할 수 있고, 도 5를 참조하여 기술한 바와 같이 변경들을 적용할 수 있다.

[0045]

동작(632)에서, 동기화 중단점 B(604)가 판정된 충돌을 해결하기 위한 충돌 해결 정책 또는 규칙이 존재하지 않는 것으로 판정하는 경우, 동기화 중단점 B(604)는 충돌하는 변경(X) 및 변경(Y)을 동기화 중단점 B(604)의 지식의 일부로서 저장할 수 있다(동작 638). 나중에, 동기화 중단점 B(604)가 다른 동기화 중단점과 동기화를 할 때, 동기화 중단점 B(604)는 충돌하는 변경들에 대한 지식을 포함한 그의 지식을 상대방 동기화 중단점으로 전송할 수 있다. 도 6C를 참조하여 기술한 바와 같이, 상대방 동기화 중단점은 충돌이 존재하는지를 판정할 수 있고 상대방 동기화 중단점이 충돌을 해결하기 위한 충돌 해결 정책 또는 규칙을 갖고 있는 경우 충돌을 해결할 수 있거나, 상대방 동기화 중단점은 충돌하는 변경들을 그의 지식의 일부로서 저장할 수 있다.

[0046]

이제 도 7을 참조하면, 변경 ID 및 지식 추적의 한 예시적인 실시예가 도시되어 있다. 도 7은 동기화 중단점(702)을 나타낸 것이다. 동기화 중단점(702)은 변경들(706)의 모음 및 지식(708)을 포함할 수 있다. 변경들(706)의 모음은 몇개의 개별적인 변경들(710)(이 예에서, X, Y 및 Z로 표시됨)을 포함할 수 있다. 도 7에 도시된 예에서, 동기화 중단점(702)의 지식의 현재 상태가 지식 벡터(712)(이 경우에, A4임)로 나타내어진다. 지식 벡터(712)는 동기화 중단점(702)의 지식(708) 모두를 나타낼 수 있다.

[0047]

또한, 도 7에는 다수의 변경 ID(714)가 나타내어져 있다. 도 7의 예에서, 동기화 중단점(702)은 변경들(710)에 대응하는 3개의 변경된 항목(716)( $I_X$ ,  $I_Y$ ,  $I_Z$ )을 포함할 수 있다. 변경 ID를 사용하여, 변경 ID(A1)을 갖는 항목( $I_X$ )이 동기화 중단점(702)에서 처음으로 만들어졌음을 알 수 있다. 변경 ID(A2)를 갖는 항목( $I_Y$ )은 변경  $I_X$  이후에 동기화 중단점(702)에서 만들어졌다. 변경 ID(A4)를 갖는 항목( $I_Z$ )은 항목( $I_Y$ )이 변경되기 전에 동기화 중단점(702)에서 만들어졌다. A3는, 도 7에 직접 나타내지는 않았지만, 이전의 변경(일례에서, 항목( $I_Z$ ))에 대한 변경(A4로 표시됨)으로 대체되는 변경 등)에 대응할 수 있다.

[0048]

변경 ID(A4)와 동기화 중단점(702)의 지식 벡터(712)(역시 A4로 표시됨) 간에는 차이가 있다. 이 예에서, 지식 벡터(A4)는 지식(708)이 변경 ID들(A4, A3, A2, A1으로 표시됨)에 대응하는 변경들을 포함한다는 것을 나타낸다. 달리 말하면, 지식 벡터는 지식 벡터와 같은 변경 ID(718)로 표현된 변경은 물론, 지식 벡터에 표현된 변경 ID(718) 이전에 행해진 동일한 동기화 중단점 ID를 갖는 모든 변경들을 포함할 수 있다. 반면에, 이 예에서는, 변경 ID(718)(A4로 표시됨)는 항목( $I_Z$ )에 행해진 변경(Z)만을 나타낸다.

[0049]

특정의 동기화 중단점에 변경이 추가될 때, 변경 ID가 예외 리스트(exception list)에 예외로서 추가될 수 있다. 예외 리스트에 포함된 동기화 중단점들로부터의 연속적인 버전의 변경들이 수락되고 또 동기화 중단점의 지식 벡터와 결합될 때 버전 번호들에 간극이 없는 경우, 예외 리스트가 지식 벡터에 병합될 수 있다. 이것에 대해서는 도 8을 참조하여 더 설명한다.

[0050]

이제 도 8을 참조하면, 2개의 동기화 중단점이 다수의 동기화 중단점들을 포함하는 토폴로지에서 동기화하는 일례가 도시되어 있다. 동기화 중단점 A(802)는 일련의 변경들(804), 지식(806), 및 지식(806)의 단축 표현인 지식 벡터(808)를 포함한다. 예로서, 동기화 중단점 A(802)의 지식 벡터(808)(A5B3C1S8)는 동기화 중단점 A의 지식(806)이 동기화 중단점 A(802)에서 제5 변경까지 행해진 변경, 동기화 중단점 B(810)에서 제3 변경까지의 지식, 동기화 중단점 C에서 제1 변경까지의 지식, 및 동기화 중단점 D에서 제8 변경까지의 지식을 포함한다는 것을 나타낸다. 도 8의 예에서, 동기화 중단점 B(810)는 일련의 변경들(812), 동기화 중단점 B의 지식(814)의 적어도 일부의 단축 표현인 지식 벡터(816)를 포함하는 지식(814), 및 예외 리스트(817)를 포함한다. 동기화 중단점 B의 지식 벡터(816)(A3B3C3D8)는 동기화 중단점 B(802)가 동기화 중단점 A(802)에 의해 행해진 제3 변경까지의 지식, 동기화 중단점 B(810)에 의해 행해진 제3 변경까지의 지식, 동기화 중단점 C에 의해 행해진 제3 변경까지의 지식, 및 동기화 중단점 D에 의해 행해진 제8 변경까지의 지식을 포함하는 지식을 갖는다는 것을 나타

낸다. 상기한 지식 벡터들은 제1 변경에서 어떤 그 다음 변경까지 동기화 중단점에 의해 행해진 변경 열거들의 연속적인 표현을 포함할 수 있다. 예외 리스트(817)는 동기화 중단점 C에 의해 행해진 제4 변경, 동기화 중단점 C에 의해 행해진 제5 변경, 및 동기화 중단점 D에 의해 행해진 제9 변경을 포함한다. 이 예에서, 동기화 중단점 C로부터의 제4 변경(C4) 및 동기화 중단점 D로부터의 제9 변경(D9)은 동일한 데이터 항목에 대한 충돌하는 변경들을 포함한다. 즉, 변경들(C4, D9)은 수락되지 않은 변경들이고, 동기화 중단점 C와 동기화 중단점 D 사이의 지연된 충돌 해결의 결과이다. 변경들(C4, D9)이 수락되지 않았기 때문에, 이들 변경이 동기화 중단점 B(810)의 지식 벡터(816)로 병합되지 않을 수 있다. 게다가, 변경(C4)이 지식 벡터(816)로 병합되지 않을 수 있기 때문에, 동기화 중단점 C로부터의 나중의 변경들(예를 들어, C5 등)도 역시 지식 벡터(816)로 병합되지 않을 수 있다.

[0051] 동기화 중단점 A(802)과 동기화 중단점 B(810)의 동기화를 시간별로 나타낸 것이 도 8에 나타내어져 있다. 시간 (1)에서, 동기화 중단점 A(802)는 동기화 중단점 A의 지식 벡터(808)로 표현될 수 있는 동기화 중단점 A의 지식(806)과 함께 동기화 요청(818)을 동기화 중단점 B(810)로 전송한다. 동기화 중단점 A(802)가 예외 리스트를 포함한 경우(이 예에서는 포함하지 않음), 그 예외 리스트가 동기화 중단점 A의 지식(806)에 포함될 것이다. 게다가, 어떤 실시예들에서, 동기화 중단점 A(802)는 별도의 동기화 요청을 전송하지 않을 수 있다. 이러한 실시예들에서, 동기화 중단점 A(802)는 동기화 요청으로서도 기능할 수 있는 지식(806)을 전송할 수 있다.

[0052] 동기화 중단점 B(810)는 시간 (2)에서 동기화 중단점 A의 지식(806)을 동기화 중단점 B에서의 변경들과 연관된 변경 ID들과 비교함으로써 지식(806)을 검사한다. 동기화 중단점 B(810)는 동기화 중단점 A(802)가 동기화 중단점 C에 의해 행해진 변경들(변경 ID들(C2, C3)로 표시됨)을 알지 못한다는 것을 발견할 수 있다. 게다가, 동기화 중단점 B(810)는 동기화 중단점 A(802)가 동기화 중단점 B의 예외 리스트(817)로부터의 변경들(C4, C5, D9)을 알지 못한다는 것을 발견할 수 있다. 따라서, 이들 변경 ID로 나타낸 변경들이 동기화 중단점 B(810)에 있는 항목들에 적용가능한 최신의 변경들이므로 간주되는 한, 동기화 중단점 B(810)는 이들 변경 ID에 대응하는 동기화 중단점 B의 변경들을 전송할 수 있다. 변경 ID가 더 이상 유효하지 않은 이전의 변경에 대응하는 경우, 그 ID에 대응하는 변경은 전송되지 않는다. 예를 들어, 버전 C3를 갖는 항목이 갱신되어 새 버전을 할당 받은 경우, C3와 연관된 변경은 동기화 중단점 B(810)에 더 이상 존재하지 않으며 동기화 중단점 A(802)로 전송되지 않는다. 그 후에 또는 이와 동시에, 시간 (3)에 나타낸 바와 같이, 동기화 중단점 B(810)는 동기화 중단점 B의 지식(814)(지식 벡터(816) 및 예외 리스트(817)로 표현될 수 있음)을 동기화 중단점 A(802)로 전송할 수 있다. 시간 (3)에서 동기화 중단점 B(810)에 의한 지식(814)의 전송은 어떤 실시예들에서 동기화 요청을 전송하는 것과 동등한 것으로 간주될 수 있다.

[0053] 시간 (4)에서, 동기화 중단점 A(802)는 동기화 중단점 B(810)에 의해 전송된 지식(814)을 동기화 중단점 A(802)에서의 변경들에 대응하는 변경 ID들과 비교함으로써 지식(814)을 검사한다. 동기화 중단점 A(802)는 동기화 중단점 B(810)가 변경 ID(A4, A5)로 표현된 변경들 또는 그 변경들에 관한 정보 어느 것도 갖지 않음을 발견한다. 따라서, (변경 ID가 더 이상 유효하지 않은 변경을 나타냄으로써 변경이 전송되지 않는 경우를 제외하고는) 동기화 중단점 A(802)는 동기화 중단점 A의 변경들(804)에 존재하는 임의의 최신 변경들 또는 이들 변경 ID에 대응하는 예외 리스트(이 예에서, 존재하지 않음)를 전송한다. 동기화 중단점 A(802)는 그 후에 모든 변경들이 전송되었음을 알려주는 메시지를 동기화 중단점 B(810)로 전송할 수 있으며, 그에 따라 동기화 중단점 A(802) 및 동기화 중단점 B(810)는, 그 변경들이 수락되고 하나 이상의 미해결된 충돌과 연관된 변경들이 아닌 경우, 이제 그의 지식 벡터(808, 816)를 최근의 변경들을 포함하도록 각각 갱신할 수 있다. 도 8에 나타낸 바와 같이, 시간 (5)에서, 동기화 중단점 A의 지식은 동기화 중단점 B의 지식 벡터와 똑같은 지식 벡터(A5B3C3D8)를 포함하고, 제5 변경 열거까지의 동기화 중단점 A(802)에 의해 행해진 모든 변경들, 제3 변경 열거까지의 동기화 중단점 B에 의해 행해진 모든 변경들, 제3 변경 열거까지의 동기화 중단점 C에 의해 행해진 모든 변경들, 및 제8 변경 열거까지의 동기화 중단점 D에 의해 행해진 모든 변경들은 물론, 예외 리스트에 저장된 변경들(C4, C5, D9)을 포함한다. 유의할 점은, 이 예에서, 변경들(C4, D9)이 동기화 중단점 C와 동기화 중단점 D 간의 충돌에 대한 정보를 포함한다는 것이다.

[0054] 도 9는 도 8에 도시된 것과 같은 동기화 이후에 지식을 갱신하는 방법을 나타낸 것이다. 구체적으로는, 도 9는 동기화 중단점 A(802)에 저장된 예외 리스트(902) 및 동기화 중단점 B(810)에 저장된 예외 리스트(903)를 사용하여 지식 벡터를 갱신하는 방법을 나타낸 것이다. 동기화 중단점들 간에 변경들이 전송될 때, 이 변경들은 그 변경과 연관된 변경 ID와 함께 전송된다. 변경이 동기화 중단점에 추가될 때, 변경 ID가 예외 리스트(902 또는 903)에 예외로서 추가될 수 있다. 도 9에서 동기화 중단점 A(802)에 대한 지식을 검사하면, 이 지식은 지식 벡터(808) 및 예외 리스트(902)(동기화 중단점 B(810)로부터 수신된 예외들(C2, C3, C4, C5)을 포함함)를 포함한

다. 이 예에서, 변경들(C4, D9)은 동일한 데이터 항목에 대한 변경에 관한 것이고, 변경(C4) 및 변경(D9)과 연관된 '알고 있는지 여부'(made-with-knowledge)는 각자의 변경이 동일한 데이터 항목에 대한 충돌하는 변경을 알지 못하고 행해졌다는 것을 알려준다. 예외 리스트(902)의 변경들(C2, C3)은, 이 예에서, 충돌하는 데이터를 갖지 않으며, 이들 변경은 동기화 중단점 A(802)의 데이터에 적용될 수 있다. 따라서, 변경들(C2, C3)은 이어서 예외 리스트(902)로부터 제거될 수 있고, 지식 벡터(808)가 갱신되어 갱신된 지식(904)을 제공할 수 있으며, 이 갱신된 지식(904)은 동기화 중단점 A(802)가 제5 변경까지의 동기화 중단점 A(802)로부터의 모든 변경들, 제3 변경까지의 동기화 중단점 B로부터의 모든 변경들, 제3 변경까지의 동기화 중단점 C로부터의 모든 변경들, 및 제8 변경까지의 동기화 중단점 D로부터의 모든 변경들을 적용하였다는 것을 알려주는 지식 벡터를 포함할 수 있다. 갱신된 지식(904)은 또한 변경들(C4, C5, D9)을 갖는 예외 리스트를 포함할 수 있다. 이 예에서, 변경들(C4, D9)은 충돌하는 변경이고, C5는 동기화 중단점 C로부터의 변경이며 적용될 수 있다.

[0055]

동기화 중단점 B(810)와 관련하여, 동기화 중단점 A(802)로부터 수신된 변경들(A4, A5)이 예외 리스트(903)에 추가될 수 있다. 변경들(C4, C5, D9)은 이미 예외 리스트에 포함되어 있을 수 있으며, 동기화 중단점 A(802)와 동기화하기 전에 동기화 중단점 B(810)가 알고 있는 지식(816)의 일부이다. 동기화 중단점 A(802)와 동기화한 후에, 갱신된 지식(906)은 지식 벡터(A5B3C3D8)를 포함할 수 있으며, 이 지식 벡터(A5B3C3D8)는 동기화 중단점 A(802)로부터의 제5 변경까지, 동기화 중단점 B(810)로부터의 제3 변경까지, 동기화 중단점 C로부터의 제3 변경까지, 및 동기화 중단점 D로부터의 제8 변경까지가 동기화 중단점 B(810)의 데이터에 적용되었다는 것을 알려준다. 지식(906)은 또한 변경들(C4, C4, D9)을 갖는 예외 리스트를 포함할 수 있다. 이 예에서, C4 및 D9은 적용되지 않은 충돌하는 변경들이고, C5는 동기화 중단점 B(810)의 데이터에 적용될 수 있는 동기화 중단점 C로부터의 제5 변경이다.

[0056]

도 8 및 도 9에 나타난 예에서, 다른 동기화 중단점으로부터의 충돌에 대한 지식이 전파될 수 있고, 동기화 중단점 A(802) 및 동기화 중단점 B(810)는 충돌에 대한 지식을 유지할 수 있으며 게다가 차후의 동기화 동안에 그 지식을 전파할 수 있다. 충돌에 대한 지식을 수신하는 동기화 중단점은 해결 정책 또는 규칙을 포함하고 있는 경우 그 충돌을 해결할 수 있다.

[0057]

도 10은 2개의 동기화 중단점이 동기화하는 것의 다른 예를 나타낸 것이다. 동기화 중단점 A(1002)는 일련의 변경들(1004), 지식(1006), 및 지식(1006)의 단축 표현인 지식 벡터(1008)를 포함하고 있다. 동기화 중단점 A(1002)의 지식 벡터(1008)(A5B3C5D9)는 동기화 중단점 A의 지식(1006)이 동기화 중단점 A(1002)에서 제5 변경까지의 행해진 변경들, 동기화 중단점 B(1010)에서 제3 변경까지의 지식, 동기화 중단점 C에서의 제5 변경까지의 지식, 및 동기화 중단점 D에서의 제9 변경까지의 지식을 포함한다는 것을 보여준다. 동기화 중단점 B(101)는, 도 10의 예에서, 일련의 변경들(1012), 동기화 중단점 B의 지식(1014)의 적어도 일부의 단축 표현인 지식 벡터(1016)를 포함하는 지식(1014), 및 예외 리스트(1017)를 포함한다. 동기화 중단점 B의 지식 벡터(1016)(A3B4C3D8)는 동기화 중단점 B가 동기화 중단점 A(1002)에 의해 행해진 제3 변경까지의 지식, 동기화 중단점 B(1010)에 의해 행해진 제4 변경까지의 지식, 동기화 중단점 C에 의해 행해진 제3 변경까지의 지식, 및 동기화 중단점 D에 의해 행해지는 제8 변경까지의 지식을 포함하는 지식을 갖는다는 것을 보여준다. 예외 리스트(1017)는 동기화 중단점 C에 의해 행해진 제4 변경, 동기화 중단점 C에 의해 행해진 제5 변경, 및 동기화 중단점 D에 의해 행해진 제9 변경을 포함한다. 이 예에서, 동기화 중단점 C로부터의 제4 변경(C4) 및 동기화 중단점 D로부터의 제9 변경(D9)은 동일한 데이터 항목에 대한 충돌하는 변경들을 포함한다. 즉, 변경들(C4, D9)은 수락되지 않은 변경들이고, 동기화 중단점 C와 동기화 중단점 D 사이의 지연된 충돌 해결의 결과이다. 변경들(C4, D9)이 수락되지 않았기 때문에, 이들 변경은 동기화 중단점 B(1010)의 지식 벡터(1016)로 병합되지 않을 수 있다. 게다가, 변경(C4)이 지식 벡터(1016)로 병합되지 않을 수 있기 때문에, 동기화 중단점 C로부터의 나중의 변경들(예를 들어, C5 등)도 역시 지식 벡터(1016)로 병합되지 않을 수 있다.

[0058]

동기화 중단점 A(1002)와 동기화 중단점 B(1010)의 동기화를 시간별로 나타낸 것이 도 10에 도시되어 있다. 시간 (1)에서, 동기화 중단점 A(1002)는 동기화 중단점 A의 지식 벡터(1008)로 표현될 수 있는 동기화 중단점 A의 지식(1006)과 함께 동기화 요청(1018)을 동기화 중단점 B(1010)로 전송할 수 있다. 동기화 중단점 A(802)가 예외 리스트를 포함하는 경우(이 예에서는 그렇지 않음), 이 예외 리스트는 동기화 중단점 A의 지식(1006)에 포함될 것이다. 앞서 언급한 바와 같이, 어떤 실시예들에서, 동기화 중단점 A(1002)는 별도의 동기화 요청을 전송하지 않을 수 있다. 이러한 실시예들에서, 동기화 중단점 A(1002)은 동기화 요청으로서도 기능할 수 있는 지식(1006)을 전송할 수 있다.

[0059]

동기화 중단점 B(1010)는, 시간 (2)에서, 동기화 중단점 A의 지식(1006)을 동기화 중단점 B(1010)에서의 변경들과 연관된 변경 ID들과 비교함으로써 지식(1006)을 검사할 수 있다. 동기화 중단점 B(1010)는 동기화 중단점

A(1002)가 동기화 종단점 B에 의해 행해진 변경(변경 ID(B4)로 표시됨)을 알지 못한다는 것을 발견할 수 있다. 따라서, 동기화 종단점 B(1010)는, 변경 ID(B4)에 대응하는 동기화 종단점 B의 변경이 동기화 종단점 B(1010)에 있는 항목에 적용가능한 최신의 변경인 것으로 간주되는 한, 그 변경을 전송한다. 변경 ID가 더 이상 유효하지 않은 이전의 변경에 대응하는 경우, 그 변경 ID에 대응하는 변경이 전송되지 않는다. 나중에 또는 이와 동시에, 시간 (3)에 나타난 바와 같이, 동기화 종단점 B(1010)는 지식 벡터(1016)로 표현될 수 있는 동기화 종단점 B의 지식(1014) 및 예외 리스트(1017)를 동기화 종단점 A(1002)로 전송할 수 있다. 시간 (3)에서 동기화 종단점 B(1010)에 의한 지식(1014)의 전송은, 어떤 실시예들에서, 동기화 요청을 전송하는 것과 동등한 것으로 간주될 수 있다.

[0060] 시간 (4)에서, 동기화 종단점 A(1002)는 동기화 종단점 B(1010)에 의해 전송된 지식(1014)을 동기화 종단점 A(802)에서의 변경들에 대응하는 변경 ID들과 비교함으로써 그 지식(1014)을 검사할 수 있다. 동기화 종단점 A(802)는 동기화 종단점 B(810)가 변경 ID들(A4, A5)에 의해 표현된 변경들이나 그 변경들에 관한 지식 어느 것도 갖지 않는다는 것을 발견할 수 있다. 따라서, (변경 ID가 더 이상 유효하지 않은 변경을 나타내고 따라서 변경이 전송되지 않을 때를 제외하고는) 동기화 종단점 A(1002)는 동기화 종단점 A의 변경들(1004)에 존재하는 임의의 최신의 변경들 또는 그 변경 ID들에 대응하는 예외 리스트(이 예에서는, 존재하지 않음)를 전송할 수 있다.

[0061] 이 예에서, 변경 ID(A5)로 나타난 변경은 변경 ID(C4, D9)로 나타난 변경들이 변경하는 것과 동일한 데이터 항목을 변경한다. 따라서, 동기화 종단점 B(1010)가 변경 ID(A5)로 나타난 변경에 대한 정보를 수신할 때, 동기화 종단점 B(1010)는 변경들(A5, C4, D9) 간에 잠재적인 충돌이 존재하는 것으로 판정할 수 있다. 그렇지만, 변경 ID(A5)에 대응하는 '알고 있는지 여부'(made-with-knowledge)는, 이 예에서, 변경 ID(A5)로 나타난 변경이 변경 ID(C4, D9)로 나타난 변경들을 알고서 행해졌다는 것을 알려준다. 따라서, 변경 ID(A5)로 나타난 변경이, 이 예에서, 변경 ID들(C4, D9)로 나타난 충돌에 대한 해결이다.

[0062] 동기화 종단점 A(1002)는 그 다음에 모든 변경들이 전송되었음을 알려주는 메시지를 동기화 종단점 B(1010)로 전송할 수 있으며, 그에 따라 동기화 종단점 A(1002) 및 동기화 종단점 B(1010)는, 그 변경들이 수락되었고 하나 이상의 미해결된 충돌들과 연관된 변경들이 아닌 경우, 이제 최근의 변경들을 포함하도록 그들의 지식 벡터(1008, 1016)를 각각 갱신할 수 있다. 도 10에 나타난 바와 같이, 시간 (5)에서, 동기화 종단점 A의 지식은 동기화 종단점 B의 지식 벡터와 똑같은 지식 벡터(A5B4C5D9)를 포함하고, 제1 변경 열거까지의 동기화 종단점 A(1002)에 의해 행해진 모든 변경들, 제4 변경 열거까지의 동기화 종단점 B에 의해 행해진 모든 변경들, 제5 변경 열거까지의 동기화 종단점 C에 의해 행해진 모든 변경들, 및 제9 변경 열거까지의 동기화 종단점 D에 의해 행해진 모든 변경들을 포함한다.

[0063] 도 11은 도 10에 도시된 것과 같은 동기화 이후에 지식을 갱신하는 방법을 나타낸 것이다. 구체적으로는, 도 11은 동기화 종단점 A(1002)에 저장된 예외 리스트(1102) 및 동기화 종단점 B(1010)에 저장된 예외 리스트(1102)를 사용하여 지식 벡터를 갱신하는 방법을 나타낸 것이다. 동기화 종단점들 간에 변경들이 전송될 때, 그 변경들은 그 변경과 연관된 변경 ID와 함께 전송될 수 있다. 변경이 동기화 종단점에 추가될 때, 변경 ID가 예외 리스트(1102)에 예외로서 추가될 수 있다. 도 11에서 동기화 종단점 A(1002)에 대한 지식을 검사할 때, 그 지식은 지식 벡터(1008)와, 동기화 종단점 B(810)로부터 수신된 예외(B4)를 포함하는 예외 리스트(1102)를 포함한다. 예외 리스트(1102)의 변경(B4)은, 이 예에서, 충돌하는 데이터를 갖지 않으며, 동기화 종단점 A(1002)의 데이터에 적용될 수 있다. 따라서, 변경(B4)이 이어서 예외 리스트(1102)로부터 제거될 수 있고, 지식 벡터(1008)가 갱신되어 갱신된 지식(1104)을 제공할 수 있으며, 이 갱신된 지식(1104)은 동기화 종단점 A(802)가 제5 변경까지의 동기화 종단점 A(802)로부터의 모든 변경들, 제4 변경까지의 동기화 종단점 B로부터의 모든 변경들, 제5 변경까지의 동기화 종단점 C로부터의 모든 변경들, 및 제9 변경까지의 동기화 종단점 D로부터의 모든 변경들을 적용하였음을 알려주는 지식 벡터를 포함할 수 있다.

[0064] 동기화 종단점 B(810)와 관련하여, 동기화 종단점 A(1002)로부터 수신된 변경들(A4, A5)이 예외 리스트(1103)에 추가될 수 있다. 변경들(C4, C5, D9)이 이미 예외 리스트에 포함되어 있을 수 있으며, 동기화 종단점 A(1002)와 동기화하기 전에 동기화 종단점 B(1010)가 알고 있는 지식(1016)의 일부일 수 있다. 동기화 종단점 A(1002)와 동기화한 후에, 갱신된 지식(1106)은 지식 벡터(A5B4C5D9)를 포함할 수 있으며, 이 지식 벡터(A5B4C5D9)는 동기화 종단점 A(802)로부터의 제5 변경까지, 동기화 종단점 B(810)로부터의 제4 변경까지, 동기화 종단점 C로부터의 제5 변경까지, 및 동기화 종단점 D로부터의 제9 변경까지가 동기화 종단점 B(810)의 데이터에 적용되었음을 알려준다. 따라서, 이 예에서, 그 변경이 미해결된 충돌을 알고서 행해졌을 때, 변경 ID들(C4, D9)로 나

타낸 충돌이 동일한 데이터 항목에 대한 변경을 수신함으로써 해결될 수 있다.

[0065]

도 12는 2개의 동기화 중단점이 동기화하는 것의 또다른 일례를 나타낸 것이다. 동기화 중단점 A(1202)는 일련의 변경들(1204), 지식(1206), 및 지식(1206)의 단축 표현인 지식 벡터(1208)를 포함한다. 동기화 중단점 A(1202)의 지식 벡터(1208)(A5B3C3D8)는 동기화 중단점 A의 지식(1206)이 동기화 중단점 A(1202)에서 제5 변경까지의 행해진 변경들, 동기화 중단점 B(1010)에서 제3 변경까지의 지식, 동기화 중단점 C에서 제3 변경까지의 지식, 및 동기화 중단점 D에서 제8 변경까지의 지식을 포함한다는 것을 보여준다. 동기화 중단점 B(1210)는, 도 12의 예에서, 일련의 변경들(1212), 동기화 중단점 B의 지식(1214)의 적어도 일부의 단축 표현인 지식 벡터(1216)를 포함하는 지식(1214), 및 예외 리스트(1217)를 포함한다. 동기화 중단점 B의 지식 벡터(1216)(A3B4C3D8)는 동기화 중단점 B가 동기화 중단점 A(1202)에 의해 행해진 제3 변경까지의 지식, 동기화 중단점 B(1210)에 의해 행해진 제4 변경까지의 지식, 동기화 중단점 C에 의해 행해진 제3 변경까지의 지식, 및 동기화 중단점 D에 의해 행해진 제8 변경까지의 지식을 포함하는 지식을 갖는다는 것을 나타낸다. 예외 리스트(1217)는 동기화 중단점 C에 의해 행해진 제4 변경, 동기화 중단점 C에 의해 행해진 제5 변경, 및 동기화 중단점 D에 의해 행해진 제9 변경을 포함한다. 이 예에서, 동기화 중단점 C로부터의 제4 변경(C4) 및 동기화 중단점 D로부터의 제9 변경(D9)은 동일한 데이터 항목에 대한 충돌하는 변경들을 포함한다. 즉, 변경들(C4, D9)은 수락되지 않은 변경들이며, 동기화 중단점 C와 동기화 중단점 D 사이의 지연된 충돌 해결의 결과이다. 변경들(C4, D9)이 수락되지 않았기 때문에, 이들 변경은 동기화 중단점 B(1210)의 지식 벡터(1216)로 병합되지 않을 수 있다. 게다가, 변경(C4)이 지식 벡터(1216)로 병합되지 않을 수 있기 때문에, 동기화 중단점 C로부터의 나중의 변경들(예를 들어, C5 등)도 역시 지식 벡터(1216)로 병합되지 않을 수 있다.

[0066]

동기화 중단점 A(1202)와 동기화 중단점 B(1210)의 동기화를 시간별로 나타낸 것이 도 12에 나타내어져 있다. 시간 (1)에서, 동기화 중단점 A(1202)는 동기화 중단점 A의 지식 벡터(1208)로 표현될 수 있는 동기화 중단점 A의 지식(1206)과 함께 동기화 요청(1218)을 동기화 중단점 B(1210)로 전송할 수 있다. 동기화 중단점 A(1202)가 예외 리스트를 포함한 경우(이 예에서는 그렇지 않음), 예외 리스트가 동기화 중단점 A의 지식(1206)에 포함될 것이다. 앞서 언급한 바와 같이, 어떤 실시예들에서, 동기화 중단점 A(1202)는 별도의 동기화 요청을 전송하지 않을 수 있다. 이러한 실시예들에서, 동기화 중단점 A(1202)는 동기화 요청으로서 역할할 수도 있는 지식(1206)을 전송할 수 있다.

[0067]

동기화 중단점 B(1210)는, 시간 (2)에서, 동기화 중단점 A의 지식(1206)을 동기화 중단점 B(1210)에서의 변경들과 연관된 변경 ID들과 비교함으로써 지식(1206)을 검사할 수 있다. 동기화 중단점 B(1210)는 동기화 중단점 A(1202)가 동기화 중단점 B에 의해 행해진 변경들(변경 ID(B4, C4, C5, D9)로 표시됨)을 알지 못하고 있다는 것을 발견할 수 있다. 따라서, 동기화 중단점 B(1210)는, 변경 ID들(B4, C4, C5, D9)에 대응하는 동기화 중단점 B의 변경들이 최신의 것으로 간주되는 한, 그 변경들을 전송할 수 있다. 변경 ID가 더 이상 유효하지 않은 이전의 변경에 대응하는 경우, 변경 ID에 대응하는 변경이 전송되지 않는다. 나중에 또는 이와 동시에, 시간 (3)에 나타낸 바와 같이, 동기화 중단점 B(1210)는 지식 벡터(1216)로 표현될 수 있는 동기화 중단점 B의 지식(1214) 및 예외 리스트(1217)를 동기화 중단점 A(1202)로 전송할 수 있다. 동기화 중단점 B(1010)에 의한 지식(1214)의 전송(시간 (3))은 어떤 실시예들에서 동기화 요청을 전송하는 것과 동등한 것으로 간주될 수 있다.

[0068]

시간 (4)에서, 동기화 중단점 A(1202)는 동기화 중단점 B(1010)에 의해 전송된 지식(1214)을 동기화 중단점 A(1202)에서의 변경들에 대응하는 변경 ID들과 비교함으로써 지식(1214)을 검사할 수 있다. 동기화 중단점 A(1202)는 동기화 중단점 B(1210)가 변경 ID들(A4, A5)로 표현된 변경들 중 어느 것도 알고 있지 않다는 것을 발견할 수 있다. 따라서, (변경 ID가 더 이상 유효하지 않은 변경을 나타내고 그에 따라 변경이 전송되지 않을 때를 제외하고는) 동기화 중단점 A(1202)는 동기화 중단점 A의 변경들(1204)에 존재하는 임의의 최신의 변경들 또는 그 변경 ID들에 대응하는 예외 리스트(이 예에서는 존재하지 않음)를 전송할 수 있다.

[0069]

이 예에서, 변경 ID(A5)로 표현된 변경은 변경 ID들(C4, D9)로 표현된 변경들이 변경하는 것과 동일한 데이터 항목을 변경한다. 따라서, 동기화 중단점 B(1210)가 변경 ID(A5)로 표현된 변경과 관련한 정보를 수신할 때, 동기화 중단점 B(1010)는 변경들(A4, C4, D9) 간에 잠재적인 충돌이 존재하는 것으로 판정할 수 있다. 이 예에서, 변경 ID(A5)에 대응하는 '알고 있는지 여부'(made-with- knowledge)는 변경 ID(A5)로 표현된 변경이 C4 및 D9로 표현된 변경들을 알지 못하고서 행해졌다는 것을 알려준다. 게다가, 변경 ID들(C4, D9)에 대응하는 '알고 있는지 여부'(made-with- knowledge)는 변경 ID들(C4, D9)로 표현된 변경들이 변경 ID(A5)로 표현된 변경을 알지 못하고서 행해졌다는 것을 알려준다. 따라서, 변경 ID(A5)로 표현된 변경은, 이 예에서, 변경 ID들(C4, D9)로 표현된 변경들과 동일한 데이터 항목을 수신하는 부가적인 충돌이다.

[0070] 동기화 중단점 A(1202)는 그 후에 모든 변경들이 전송되었음을 알려주는 메시지를 동기화 중단점 B(1210)로 전송할 수 있으며, 따라서, 그 변경들이 수락되었고 하나 이상의 미해결된 충돌과 연관된 변경들이 아닌 경우, 동기화 중단점 A(1202) 및 동기화 중단점 B(1210)가 이제 최근의 변경들을 포함하도록 자기의 지식 벡터(1208, 1216)를 각각 갱신할 수 있다. 도 12에 도시된 바와 같이, 시간 (5)에서, 동기화 중단점 A의 지식은 동기화 중단점 B의 지식 벡터와 똑같은 지식 벡터(A4B4C3D8)를 포함하고, 제4 변경 열거까지의 동기화 중단점 A(1202)에 의해 행해진 모든 변경들, 제4 변경 열거까지의 동기화 중단점 B에 의해 행해진 모든 변경들, 제3 변경 열거까지의 동기화 중단점 C에 의해 행해진 모든 변경들, 및 제8 변경 열거까지의 동기화 중단점 D에 의해 행해진 모든 변경들을 포함한다. 게다가, 동기화 중단점 A(1202) 및 동기화 중단점 B(1210)는 그 각자의 지식에 예외 리스트(A5, C4, C5, D9)를 포함한다. 이 예에서, A5, C4 및 D9는 동일한 데이터 항목에 대한 미해결된 충돌하는 변경들을 나타낸다. 이전에 동기화 중단점 A(1202)에 의해 수락되었던 변경 ID(A4)에 대응하는 변경들은 충돌 관정으로 인해 수락되지 않을 수 있고, 동기화 중단점 A의 지식(1206)에 포함된 예외 리스트에 추가될 수 있다.

[0071] 도 13은 도 12에 도시된 것과 같은 동기화 이후에 지식을 갱신하는 방법을 나타낸 것이다. 구체적으로는, 도 13은 동기화 중단점 A(1202)에 저장된 예외 리스트(1302) 및 동기화 중단점 B(1210)에 저장된 예외 리스트(1303)를 사용하여 지식 벡터를 갱신하는 방법을 나타낸 것이다. 동기화 중단점들 간에 변경들이 전송될 때, 그 변경들은 그 변경과 연관된 변경 ID와 함께 전송될 수 있다. 변경이 동기화 중단점 A(1202)와 같은 동기화 중단점에 추가될 때, 변경 ID가 예외 리스트(1302)에 예외로서 추가될 수 있다. 도 13에서 동기화 중단점 A(1202)에 대한 지식을 검사할 때, 지식은 지식 벡터(1208)와, 동기화 중단점 B(1210)로부터 수신된 예외들(B4, C4, C5, D9)을 포함할 수 있는 예외 리스트(1302)를 포함할 수 있다. 예외 리스트(1302)의 변경 ID(B4)로 표현된 변경은, 이 예에서, 충돌하는 변경들을 갖지 않으며, 동기화 중단점 A(1202)의 데이터에 적용될 수 있다. 그렇지만, 변경 ID들(C4, D9)과 연관된 변경들은 충돌하는 데이터를 가지며, 동기화 중단점 A(1202)의 데이터에 적용될 수 없다. 따라서, 변경 ID(B4)는 이어서 예외 리스트(1302)로부터 제거될 수 있고, 지식 벡터(1208)는 갱신되어 갱신된 지식(1304)을 제공할 수 있으며, 이 갱신된 지식(1304)은 동기화 중단점 A(1202)가 제4 변경까지의 동기화 중단점 A(1202)로부터의 변경들, 제4 변경까지의 동기화 중단점 B로부터의 모든 변경들, 제3 변경까지의 동기화 중단점 C로부터의 모든 변경들, 제8 변경까지의 동기화 중단점 D로부터의 모든 변경들을 적용했다는 것을 알려주는 지식 벡터를 포함할 수 있다. 지식(1304)은 또한 예외들(A4, C4, C5, D9)을 포함할 수 있으며, 여기서 변경 ID들(A5, C4, D9)로 표현된 변경들이 동일한 데이터 항목에 대한 충돌하는 변경들을 갖는다. 이 예에서, 변경 ID(A5)와 연관된 변경이 충돌의 일부이기 때문에, 이 변경이 동기화 중단점 A(1202)에 의해 수락되지 않을 수 있으며 그 변경을 나타내는 변경 ID(A5)가 지식(1304)에 포함된 예외 리스트에 추가될 수 있다.

[0072] 동기화 중단점 B(1210)와 관련하여, 동기화 중단점 A(1202)로부터 수신된 변경들(A4, A5)이 예외 리스트(1303)에 추가될 수 있다. 변경들(C4, C5, D9)은 이미 예외 리스트에 포함되어 있을 수 있으며, 동기화 중단점 A(1202)와 동기화하기 전에 동기화 중단점 B(1210)가 알고 있는 지식(1216)의 일부일 수 있다. 동기화 중단점 A(1202)와 동기화한 후에, 갱신된 지식(1306)은 지식 벡터(A4B4C3D8)를 포함할 수 있으며, 이 지식 벡터(A4B4C3D8)는 동기화 중단점 A(1202)로부터의 제4 변경까지, 동기화 중단점 B(1210)로부터의 제4 변경까지, 동기화 중단점 C로부터의 제3 변경까지, 및 동기화 중단점 D로부터의 제8 변경까지가 동기화 중단점 B(1210)의 데이터에 적용되었다는 것을 알려준다. 갱신된 지식(1306)은 또한 변경 ID들(A5, C4, C5, D9)에 대응하는 예외들을 포함할 수 있으며, 여기서 변경 ID들(A5, C4, D9)로 표현된 변경들이 동일한 데이터 항목에 대한 충돌하는 변경이다. 따라서, 이 예에서, 동일한 데이터 항목에 대한 다수의 충돌하는 변경들을 알려주는 정보가 다른 동기화 중단점들로 전파될 수 있다.

[0073] **결론**

[0074] 본 발명이 구조적 특징들 및/또는 방법적 동작들과 관련하여 기술되어 있지만, 첨부된 청구항들에서의 발명 대상이 상기한 특징의 특징들 또는 동작들로 꼭 제한되는 것은 아님을 잘 알 것이다. 오히려, 상기한 특징의 특징들 및 동작들은 청구항들을 구현하는 예시적인 형태로서 개시된 것이다.

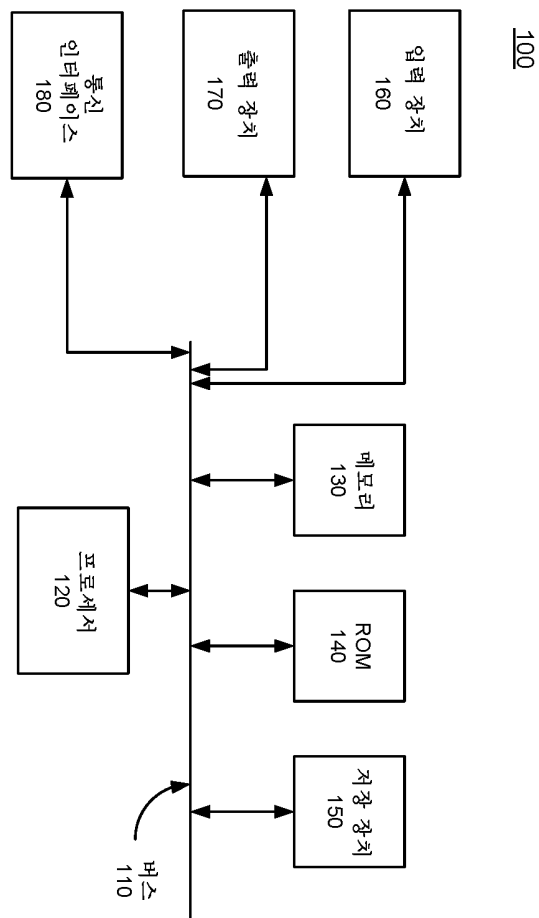
[0075] 이상의 설명이 구체적인 상세를 포함할 수 있지만, 이들 상세가 결코 청구항들을 제한하는 것으로 해석되어서는 안된다. 기술된 실시예들의 다른 구성들은 본 발명의 범위의 일부이다. 게다가, 본 발명에 따른 구현예들은 기술된 것보다 더 많은 또는 더 적은 동작들을 가질 수 있거나, 도시된 것과 다른 순서로 동작들을 구현할 수 있다. 그에 따라, 주어진 특징의 예가 아니라, 첨부된 청구항들 및 이들의 법적 등가물만이 본 발명을 한정하는 것으로 보아야 한다.

### 도면의 간단한 설명

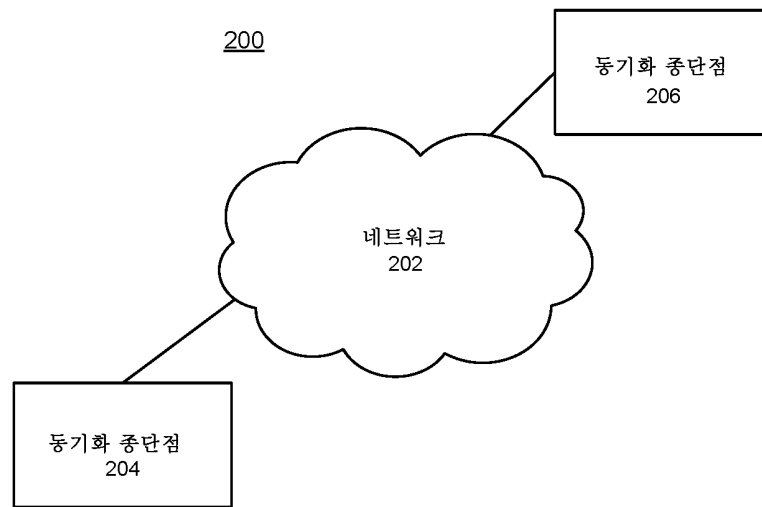
- [0006] 도 1은 동기화 중단점이 구현될 수 있는 처리 장치의 기능 블록도.
- [0007] 도 2는 동기화 중단점들의 동기화 커뮤니티에 대한 예시적인 동작 환경을 나타낸 도면.
- [0008] 도 3은 동기화 커뮤니티의 예시적인 토폴로지를 나타낸 도면.
- [0009] 도 4는 본 발명에 따른 실시예에서 동기화 중단점에서 변경들이 어떻게 관리될 수 있는지의 일례를 시간별로 나타낸 도면.
- [0010] 도 5는 본 발명에 따른 실시예에서 동기화 동안에 지식을 사용하여 변경들을 열거하는 것의 일례를 시간별로 나타낸 도면.
- [0011] 도 6A 내지 도 6C는 동기화 데이터 충돌을 검출하는 예시적인 실시예를 나타낸 도면.
- [0012] 도 7은 동기화 동작 동안에 변경 ID들 및 지식을 추적하는 것의 예시적인 실시예를 나타낸 도면.
- [0013] 도 8, 도 10 및 도 12는 본 발명에 따른 예시적인 실시예들에서 예외 리스트를 사용하는 동기화를 나타낸 도면.
- [0014] 도 9, 도 11 및 도 13은 각각 도 8, 도 10 및 도 12에 도시된 것과 같은 동기화 이후에 지식을 업데이트하는 예시적인 방법을 나타낸 도면.

### 도면

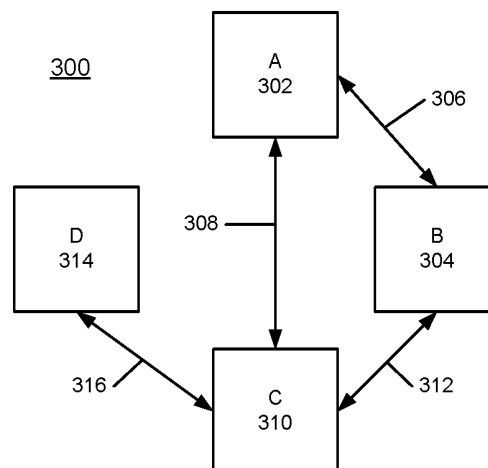
#### 도면1



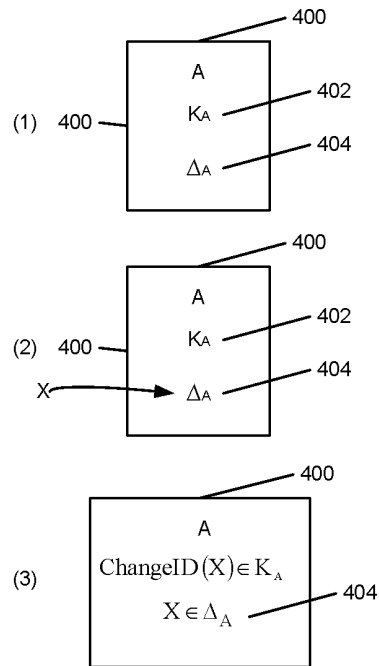
도면2



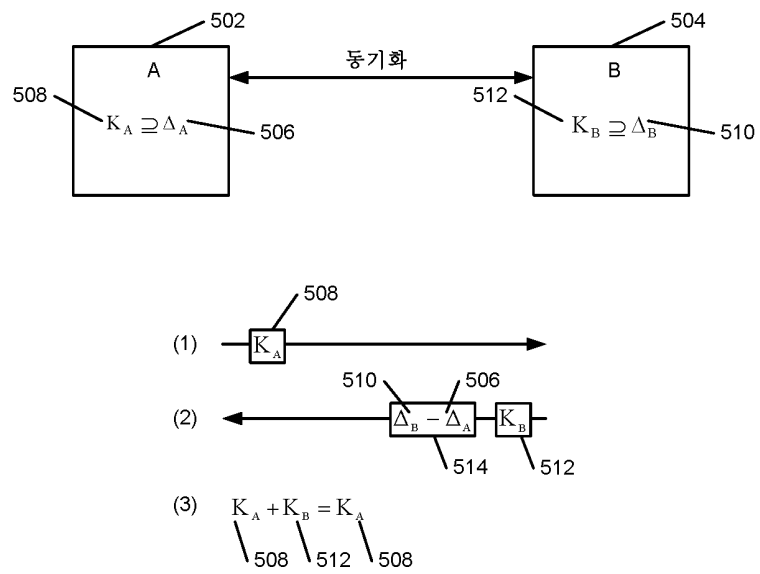
도면3



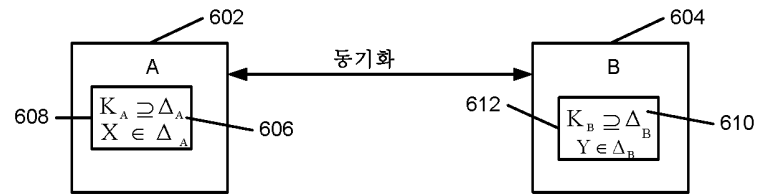
도면4



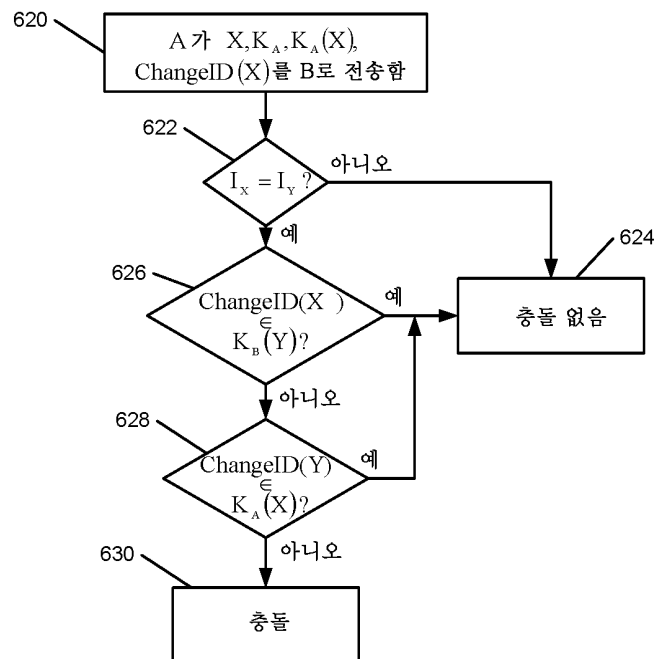
도면5



도면6A

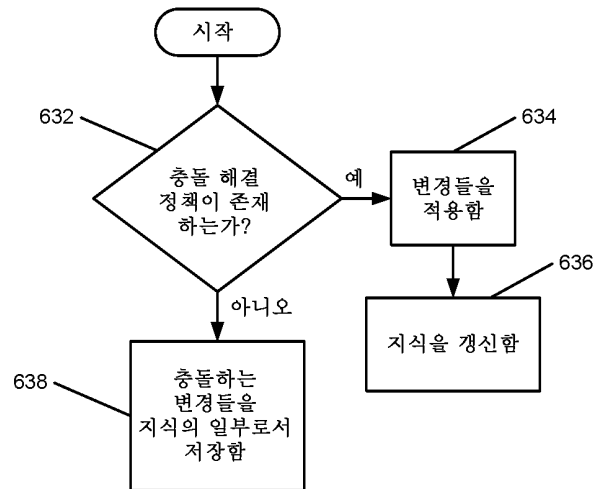


도면6B



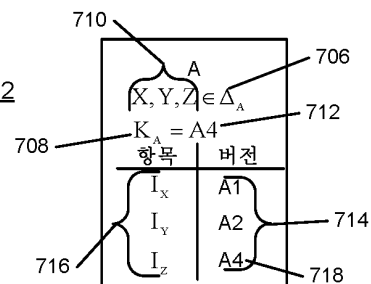
도면6C

630

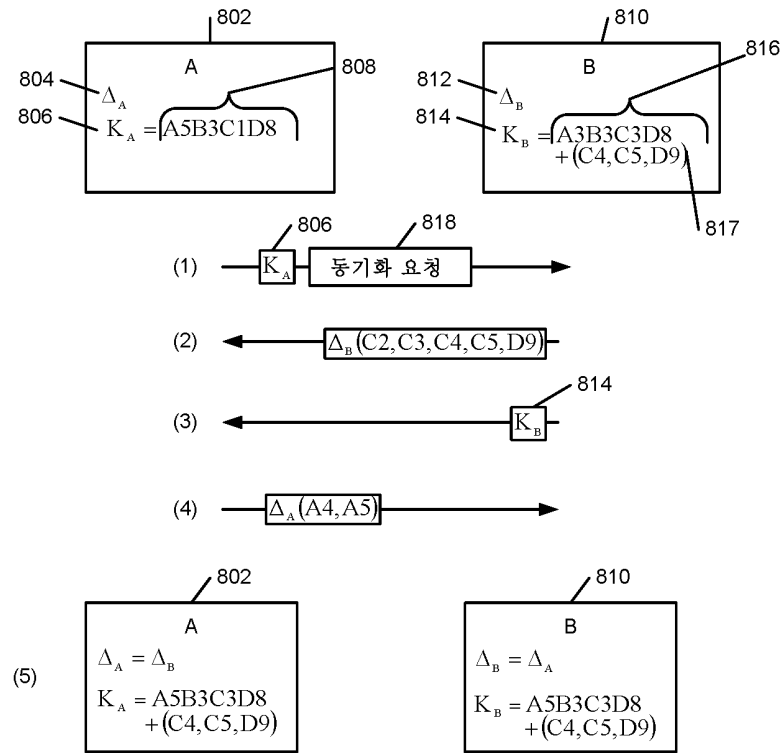


도면7

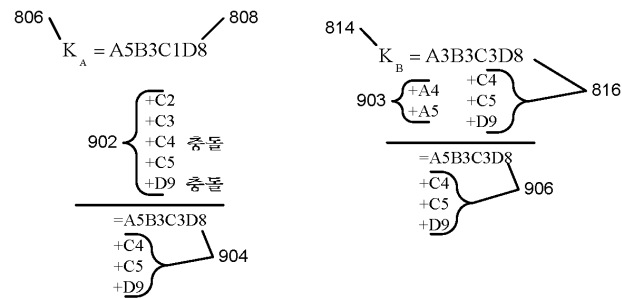
702



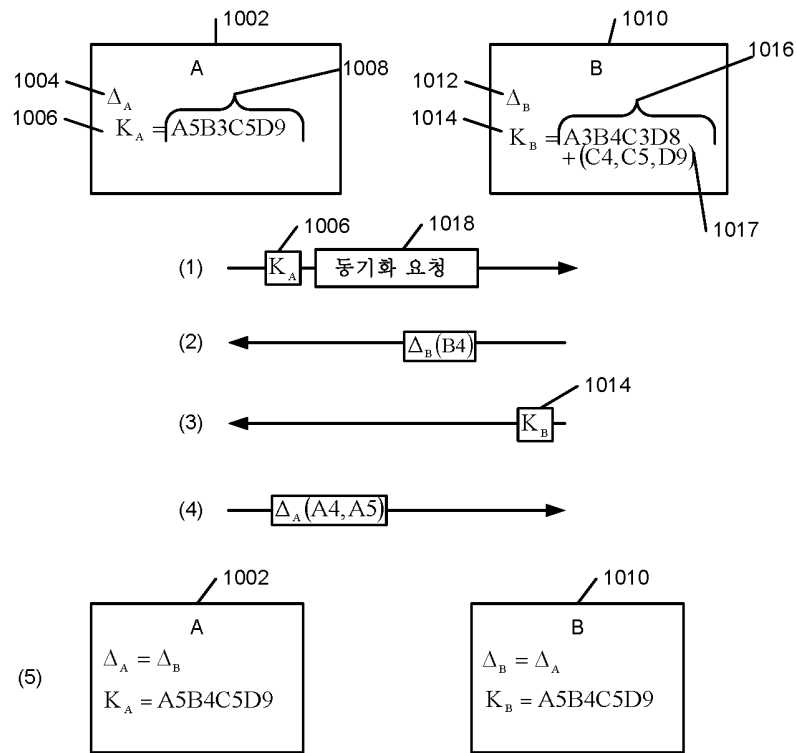
도면8



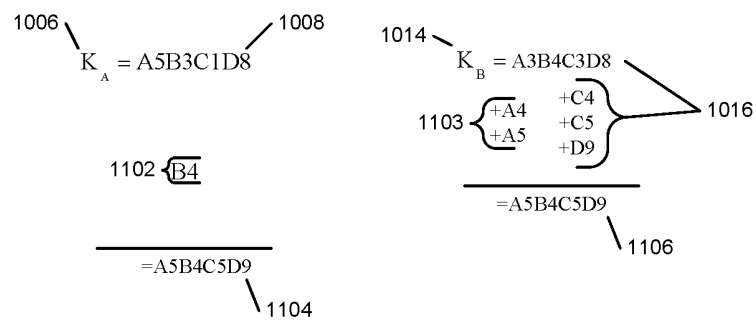
도면9



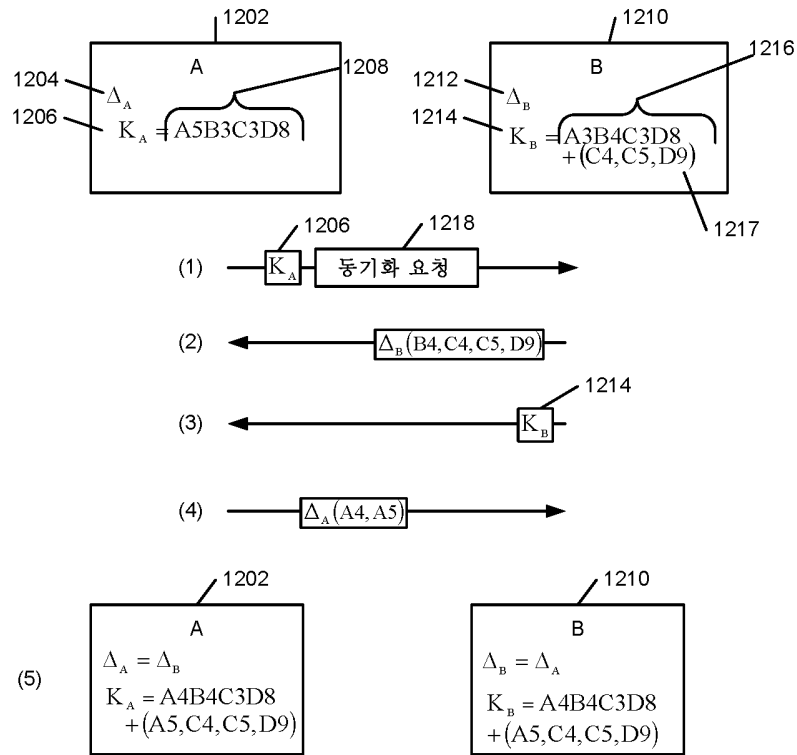
도면10



도면11



도면12



도면13

