(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2009년07월06일
(11) 등록번호 10-0906262
(24) 등록일자 2009년06월29일

(51) Int. Cl.
G06F 12/00 (2006.01)
(21) 출원번호 10-2004-0005106
(22) 출원일자 2004년01월27일
심사청구일자 2007년04월30일
(65) 공개번호 10-2004-0069282
(43) 공개일자 2004년08월05일
(30) 우선권주장
10/352,599 2003년01월28일 미국(US)
(56) 선행기술조사문헌
EP1113413 A2*
US05706462 A1*
US6128627 A
JP2001043204 A
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
브라운데이비드씨.
미국98052워싱턴주레드몬드에이피티이-322엔이40 번스트리트17525
리에노브마이크헤일브이.
미국98033
워싱턴주키르클랜드엔이넘버에이-111122 번씨티9205
버드마이클엠.
미국98007워싱턴주벨리뷰피엠비32914150엔이20
(74) 대리인
백만기, 이중희, 주성민

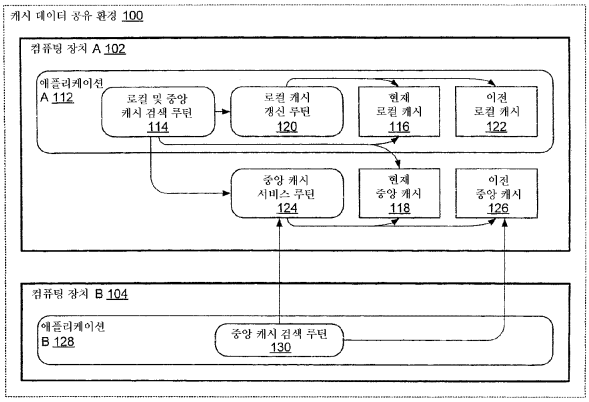
전체 청구항 수 : 총 37 항 심사관 : 권오성

(54) 원자적으로 갱신되는 중앙 캐시 메모리를 위한 방법 및시스템

(57) 요 약

본 발명은 잠금으로 인한 지연 오버헤드(overhead) 없이 갱신(update)되는 중앙 캐시를 제공한다. 갱신은 중도에 인터럽트(interrupt)될 수 없도록 되어 있다는 점에서 "원자적(atomic)"이다. 애플리케이션은 항상 중앙 캐시의 데이터를 자유로이 판독할 수 있으며, 레퍼런스 테이블(reference table)을 통해 데이터를 액세스한다. 애플리케이션은 캐시를 직접 갱신하지 않으며, 대신 갱신 요청을 서비스 루틴으로 보낸다. 캐시를 갱신하기 위해서, 캐시 서비스 루틴은 두 단계로 진행된다. 제1 단계에서, 캐시 서비스 루틴은 레퍼런스 테이블의 갱신 없이 새로운 데이터를 준비하여 이를 캐시에 추가한다. 제1 단계 동안, 캐시를 액세스하는 애플리케이션은 새로운 데이터를 "보는 것"이 불가능한 바, 이는 레퍼런스 테이블이 아직 갱신되지 않았기 때문이다. 제1 단계가 완료되면, 서비스 루틴은 갱신 프로세스의 제2 단계, 즉 레퍼런스 테이블의 원자적 갱신을 수행한다. 이러한 2단계 갱신은 캐시가 항상 일관된 상태로 있도록 해 준다.

대 표 도



특허청구의 범위

청구항 1

애플리케이션 프로그램, 중앙 캐시 및 상기 애플리케이션 프로그램과는 별개인 캐시 서비스 루틴(cache service routine)을 갖는 컴퓨팅 환경에서, 상기 캐시 서비스 루틴이 상기 중앙 캐시에 관심 데이터(data of interest)를 추가하려는 시도를 하는 방법으로서,

상기 중앙 캐시에 상기 관심 데이터를 추가해 달라는 요청을 상기 애플리케이션으로부터 수신하는 단계와,

상기 중앙 캐시에 상기 관심 데이터를 추가할지 여부를 결정하는 단계와,

상기 관심 데이터를 추가하라는 결정에 따라 상기 중앙 캐시에 상기 관심 데이터를 추가하는 단계를 포함하고,

상기 중앙 캐시에 상기 관심 데이터를 추가하는 상기 단계는

첫째로, 상기 중앙 캐시에 상기 관심 데이터를 저장하는 단계와,

둘째로, 상기 관심 데이터에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 단계를 포함하고,

상기 관심 데이터에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 상기 단계는 원자적 동작(atomic operation)인 방법.

청구항 2

제1항에 있어서, 상기 애플리케이션 프로그램은 제1 컴퓨팅 장치 상에서 실행되고, 상기 캐시 서비스 루틴은 제2 컴퓨팅 장치 상에서 실행되며, 상기 제1 및 제2 컴퓨팅 장치는 서로 별개인 방법.

청구항 3

제2항에 있어서, 상기 애플리케이션 프로그램으로부터 요청을 수신하는 상기 단계는, 직렬 통신 라인, 병렬 통신 라인, LAN(Local Area Network), 인트라넷(intranet), 모뎀 링크 및 인터넷으로 이루어지는 그룹으로부터 선택된 통신 매체를 통해 수신하는 단계를 포함하는 방법.

청구항 4

제1항에 있어서, 상기 관심 데이터는 폰트 글리프(font glyph)를 포함하는 방법.

청구항 5

제1항에 있어서, 상기 중앙 캐시에 상기 관심 데이터를 추가할지 여부를 결정하는 상기 단계는 적어도 부분적으로

상기 관심 데이터가 상기 중앙 캐시에 들어맞을지 여부를 결정하는 단계와,

상기 관심 데이터가 상기 중앙 캐시에서 이미 존재하는지 여부를 결정하는 단계와,

상기 애플리케이션 프로그램으로부터 또는 다른 애플리케이션 프로그램으로부터 수신된 요청 및 다른 요청에 관한 통계를 수집하는 단계와,

상기 애플리케이션으로부터 통계를 수신하는 단계

로 이루어지는 그룹으로부터 선택된 동작에 기초하는 방법.

청구항 6

제5항에 있어서, 통계를 수집하는 상기 단계는, 상기 관심 데이터에 대한 요청의 개수, 상기 관심 데이터에 대한 요청의 시간적 빈도, 상기 관심 데이터의 크기 및 상기 관심 데이터에 대한 요청의 출처로 이루어지는 그룹으로부터 선택된 통계를 수집하는 단계를 포함하는 방법.

청구항 7

삭제

청구항 8

삭제

청구항 9

제1항에 있어서, 상기 중앙 캐시에 상기 관심 데이터를 추가하는 상기 단계는 상기 중앙 캐시의 레퍼런스 테이블에 있는 항목을 선택하는 단계를 포함하고, 상기 선택은 적어도 부분적으로 상기 관심 데이터의 해시(hash)에 기초하는 방법.

청구항 10

제9항에 있어서, 상기 선택된 항목을 검사한 결과 상기 선택된 항목이 상기 중앙 캐시에 있는 데이터 요소를 참조하지 않는 것으로 판명된 경우,

상기 관심 데이터를 상기 중앙 캐시에 있는 새로운 데이터 요소에 배치하는 단계와,

상기 관심 데이터에 대한 레퍼런스를 상기 레퍼런스 테이블의 상기 선택된 항목에 추가하는 단계

를 더 포함하는 방법.

청구항 11

제10항에 있어서, 상기 관심 데이터에 대한 레퍼런스를 추가하는 상기 단계는 상기 새로운 데이터 요소에 대한 정수 오프셋(offset)을 계산하는 단계를 포함하는 방법.

청구항 12

제9항에 있어서, 상기 선택된 항목을 검사한 결과 상기 선택된 항목이 상기 중앙 캐시에 있는 데이터 요소를 참조하는 것으로 판명된 경우,

상기 데이터 요소를 검사한 결과 상기 데이터 요소가 다른 데이터 요소를 참조하는 것으로 판명된 경우, 다른 데이터 요소를 참조하지 않는 데이터 요소에 도달할 때까지, 다른 데이터 요소를 검사하는 데이터 요소 검사 단계를 반복하는 단계와,

다른 데이터 요소를 참조하지 않는 데이터 요소에 도달한 경우, 상기 관심 데이터를 상기 중앙 캐시에 있는 새로운 데이터 요소에 배치하는 단계 및 상기 관심 데이터에 대한 레퍼런스를 상기 데이터 요소에 추가하는 단계

를 더 포함하는 방법.

청구항 13

제12항에 있어서, 상기 관심 데이터에 대한 레퍼런스를 추가하는 상기 단계는 상기 새로운 데이터 요소에 대한 정수 오프셋을 계산하는 단계를 포함하는 방법.

청구항 14

제1항에 있어서,

새로운 중앙 캐시를 생성할지 여부를 결정하는 단계와,

새로운 중앙 캐시를 생성하라는 결정에 따라 새로운 중앙 캐시를 생성하는 단계

를 더 포함하는 방법.

청구항 15

제14항에 있어서, 새로운 중앙 캐시를 생성할지 여부를 결정하는 상기 단계는 적어도 부분적으로

상기 관심 데이터가 상기 중앙 캐시에 들어맞을지 여부를 결정하는 단계와,

상기 중앙 캐시의 레퍼런스 테이블이 가득 차있는지 여부를 결정하는 단계와,

상기 중앙 캐시에 있는 데이터가 얼마나 최근에 사용되었는지를 결정하는 단계

로 이루어지는 그룹으로부터 선택된 동작에 기초하는 방법.

청구항 16

제14항에 있어서, 새로운 중앙 캐시를 생성하는 상기 단계는

새로운 중앙 캐시를 생성하는 단계와,

상기 새로운 중앙 캐시에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 단계와,

상기 중앙 캐시를 폐기된 것으로 표지하는 단계

를 포함하는 방법.

청구항 17

제16항에 있어서, 상기 폐기된 중앙 캐시로부터 선택된 데이터 요소를 상기 새로운 중앙 캐시에 배치하는 단계를 더 포함하고, 상기 선택은 적어도 부분적으로, 상기 폐기된 중앙 캐시에 있는 데이터 요소의 사용에 관하여 수집된 통계에 적용된 휴리스틱에 기초하는 방법.

청구항 18

제17항에 있어서, 상기 휴리스틱은 가장 최근에 사용됨(most recently used), 가장 자주 사용됨(most often used) 및 순환 순서(round robin)로 이루어지는 그룹으로부터 선택되는 방법.

청구항 19

제16항에 있어서, 상기 폐기된 중앙 캐시를 참조하는 애플리케이션 프로그램이 존재하지 않는 경우 상기 폐기된 중앙 캐시가 자동으로 삭제되도록 상기 중앙 캐시에 표지하는 단계를 더 포함하는 방법.

청구항 20

애플리케이션 프로그램과는 별개인 캐시 서비스 루틴이 중앙 캐시에 관심 데이터를 추가하려는 시도를 하는 방법을 수행하기 위한 명령어를 포함하는 컴퓨터로 판독 가능한 기록 매체로서, 상기 방법은

상기 중앙 캐시에 상기 관심 데이터를 추가해 달라는 요청을 상기 애플리케이션으로부터 수신하는 단계와,

상기 중앙 캐시에 상기 관심 데이터를 추가할지 여부를 결정하는 단계와,

상기 관심 데이터를 추가하라는 결정에 따라 상기 중앙 캐시에 상기 관심 데이터를 추가하는 단계를 포함하고,

상기 중앙 캐시에 상기 관심 데이터를 추가하는 상기 단계는

첫째로, 상기 중앙 캐시에 상기 관심 데이터를 저장하는 단계와,

둘째로, 상기 관심 데이터에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 단계를 포함하고,

상기 관심 데이터에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 상기 단계는 원자적 동작(atomic operation)인 컴퓨터로 판독 가능한 기록 매체.

청구항 21

애플리케이션 프로그램, 중앙 캐시 및 상기 애플리케이션 프로그램과는 별개인 캐시 서비스 루틴을 갖는 컴퓨팅 환경에서, 상기 애플리케이션 프로그램이 관심 데이터를 액세스하려는 시도를 하는 방법으로서,

상기 중앙 캐시가 폐기된 것으로 표지되었는지를 파악하기 위해 상기 중앙 캐시를 체크하는 단계와,

상기 중앙 캐시가 폐기된 것으로 표지된 경우, 다른 중앙 캐시를 찾기 위해 상기 폐기된 중앙 캐시에 있는 레퍼런스를 사용하는 단계와,

상기 중앙 캐시가 폐기된 것으로 표지되지 않은 경우,

상기 중앙 캐시에서 상기 관심 데이터를 검색하는 단계와,

상기 중앙 캐시에서 상기 관심 데이터가 발견되었는지를 결정하는 단계와,

상기 중앙 캐시에서 상기 관심 데이터가 발견된 경우, 상기 중앙 캐시에 있는 상기 관심 데이터를 액세스하는 단계와,

상기 중앙 캐시에서 상기 관심 데이터가 발견되지 않은 경우, 상기 중앙 캐시가 아닌 다른 장소에서 상기 관심 데이터를 액세스하려는 시도를 하는 단계 및 상기 중앙 캐시에 상기 관심 데이터를 추가해 달라는 요청을 상기 캐시 서비스 루틴에 보내는 단계

를 포함하는 방법.

청구항 22

제21항에 있어서, 상기 애플리케이션 프로그램은 제1 컴퓨팅 장치 상에서 실행되고, 상기 캐시 서비스 루틴은 제2 컴퓨팅 장치 상에서 실행되며, 상기 제1 및 제2 컴퓨팅 장치는 서로 별개인 방법.

청구항 23

제22항에 있어서, 요청을 상기 캐시 서비스 루틴에 보내는 상기 단계는, 직렬 통신 라인, 병렬 통신 라인, LAN, 인트라넷, 모뎀 링크 및 인터넷으로 이루어지는 그룹으로부터 선택된 통신 매체를 통해 보내는 단계를 포함하는 방법.

청구항 24

제21항에 있어서, 상기 관심 데이터는 폰트 글리프를 포함하는 방법.

청구항 25

제21항에 있어서, 상기 중앙 캐시에서 상기 관심 데이터를 검색하는 상기 단계는 상기 중앙 캐시의 레퍼런스 테이블에 있는 항목을 선택하는 단계를 포함하고, 상기 선택은 적어도 부분적으로 상기 관심 데이터의 해시에 기초하는 방법.

청구항 26

제25항에 있어서, 상기 선택된 항목을 검사한 결과 상기 선택된 항목이 상기 중앙 캐시에 있는 데이터 요소를 참조하지 않는 것으로 판명된 경우, 상기 관심 데이터가 상기 중앙 캐시에 존재하지 않는 것으로 결정하는 단계를 더 포함하는 방법.

청구항 27

제25항에 있어서, 상기 선택된 항목을 검사한 결과 상기 선택된 항목이 상기 중앙 캐시에 있는 데이터 요소를 참조하는 것으로 판명된 경우 및 상기 데이터 요소를 검사한 결과 상기 데이터 요소가 상기 관심 데이터를 포함하는 것으로 판명된 경우, 상기 관심 데이터가 상기 중앙 캐시에 존재하는 것으로 결정하는 단계를 더 포함하는 방법.

청구항 28

제27항에 있어서, 상기 선택된 항목은 정수 오프셋에 의해 상기 중앙 캐시에 있는 데이터 요소를 참조하는 방법.

청구항 29

제25항에 있어서, 상기 선택된 항목을 검사한 결과 상기 선택된 항목이 상기 중앙 캐시에 있는 데이터 요소를 참조하는 것으로 판명된 경우 및 상기 데이터 요소를 검사한 결과 상기 데이터 요소가 상기 관심 데이터를 포함하지 않는 것으로 판명된 경우,

상기 데이터 요소가 다른 데이터 요소를 참조하지 않는 경우, 상기 관심 데이터가 상기 중앙 캐시에 존재하지 않는 것으로 결정하는 단계와,

상기 데이터 요소가 다른 데이터 요소를 참조하는 경우, 상기 관심 데이터를 포함하는 데이터 요소에 도달할 때까지 또는 다른 데이터 요소를 참조하지 않는 데이터 요소에 도달할 때까지, 다른 데이터 요소를 검사하는 데이

터 요소 검사 단계를 반복하는 단계

를 더 포함하는 방법.

청구항 30

제21항에 있어서, 상기 중앙 캐시가 아닌 다른 곳에서 상기 관심 데이터를 액세스하려는 시도를 하는 상기 단계는,

상기 애플리케이션 프로그램의 로컬 캐시에서 상기 관심 데이터를 검색하는 단계와,

상기 로컬 캐시에서 상기 관심 데이터가 발견된 경우, 상기 로컬 캐시에 있는 상기 관심 데이터를 액세스하는 단계와,

상기 로컬 캐시에서 상기 관심 데이터가 발견되지 않은 경우, 상기 관심 데이터를 생성하는 루틴을 호출하는 단계

를 포함하는 방법.

청구항 31

제30항에 있어서, 상기 로컬 캐시에서 상기 관심 데이터가 발견되지 않은 경우, 생성된 상기 관심 데이터를 상기 로컬 캐시에 추가하는 단계를 더 포함하는 방법.

청구항 32

제30항에 있어서, 상기 로컬 캐시에서 상기 관심 데이터가 발견되지 않은 경우, 생성된 상기 관심 데이터를 상기 중앙 캐시에 추가해 달라는 요청을 상기 캐시 서비스 루틴에 보내는 단계를 더 포함하는 방법.

청구항 33

제21항에 있어서, 요청을 상기 캐시 서비스 루틴에 보내는 상기 단계는

상기 요청을 저장하는 단계와,

하나 또는 그 이상의 다른 요청을 저장하는 단계와,

상기 저장된 요청을 상기 캐시 서비스 루틴에 보내는 단계

를 포함하는 방법.

청구항 34

삭제

청구항 35

제21항에 있어서,

상기 애플리케이션 프로그램의 상기 중앙 캐시 사용에 관한 통계를 수집하는 단계와,

상기 수집된 통계를 상기 캐시 서비스 루틴에 보내는 단계

를 더 포함하는 방법.

청구항 36

애플리케이션 프로그램이 관심 데이터를 액세스하려는 시도를 하는 방법을 수행하기 위한 명령어를 포함하는 컴퓨터로 판독 가능한 기록 매체로서, 상기 방법은

중앙 캐시가 폐기된 것으로 표지되었는지를 파악하기 위해 상기 중앙 캐시를 체크하는 단계와,

상기 중앙 캐시가 폐기된 것으로 표지된 경우, 다른 중앙 캐시를 찾기 위해 상기 폐기된 중앙 캐시에 있는 레퍼런스를 사용하는 단계와,

상기 중앙 캐시가 폐기된 것으로 표지되지 않은 경우,

상기 중앙 캐시에서 상기 관심 데이터를 검색하는 단계와,

상기 중앙 캐시에서 상기 관심 데이터가 발견되었는지를 결정하는 단계와,

상기 중앙 캐시에서 상기 관심 데이터가 발견된 경우, 상기 중앙 캐시에 있는 상기 관심 데이터를 액세스하는 단계와,

상기 중앙 캐시에서 상기 관심 데이터가 발견되지 않은 경우, 상기 중앙 캐시가 아닌 다른 곳에서 상기 관심 데이터를 액세스하려는 시도를 하는 단계 및 상기 중앙 캐시에 상기 관심 데이터를 추가해 달라는 요청을 상기 애플리케이션 프로그램과 별개인 캐시 서비스 루틴에 보내는 단계

를 포함하는 컴퓨터로 판독 가능한 기록 매체.

청구항 37

삭제

청구항 38

삭제

청구항 39

삭제

청구항 40

삭제

청구항 41

삭제

청구항 42

캐시의 데이터 구조를 저장한 컴퓨터로 판독 가능한 기록 매체로서, 상기 캐시 데이터 구조는

헤더를 나타내는 데이터를 포함하는 제1 데이터 필드와,

복수의 데이터 요소에 대한 레퍼런스의 테이블을 나타내는 데이터를 포함하는 제2 데이터 필드와,

상기 복수의 데이터 요소를 나타내는 데이터를 포함하는 제3 데이터 필드

를 포함하고,

상기 제2 데이터 필드는 상기 제3 데이터 필드에 있는 상기 데이터 요소들에 대한 정수 오프셋을 포함하고,

상기 제1 데이터 필드는

상기 캐시가 폐기된 것임을 표지하기 위한 플래그를 나타내는 데이터를 포함하는 제4 데이터 필드와,

다른 캐시에 대한 레퍼런스를 나타내는 데이터를 포함하는 제5 데이터 필드

를 포함하는 컴퓨터로 판독 가능한 기록 매체.

청구항 43

삭제

청구항 44

제42항에 있어서, 상기 제3 데이터 필드에 있는 하나의 데이터 요소는

상기 제3 데이터 필드에 있는 다른 데이터 요소에 대한 정수 오프셋을 나타내는 데이터를 포함하는 제6 데이터

필드와,

데이터 요소 유형 식별자를 나타내는 데이터를 포함하는 제7 데이터 필드와,

데이터 요소에 특유한 데이터를 나타내는 데이터를 포함하는 제8 데이터 필드

를 포함하는 컴퓨터로 판독 가능한 기록 매체.

청구항 45

제44항에 있어서, 상기 데이터 요소에 특유한 데이터는 폰트 글리프를 포함하는 컴퓨터로 판독 가능한 기록 매체.

청구항 46

중앙 캐시에 데이터를 저장하고, 상기 중앙 캐시에 저장된 데이터를 액세스하는 시스템으로서,

데이터를 저장하기 위한 중앙 캐시와,

상기 중앙 캐시에서 관심 데이터를 액세스하려는 시도를 하고, 상기 중앙 캐시에서 상기 관심 데이터를 발견하지 못한 경우에는 상기 관심 데이터를 상기 중앙 캐시에 추가해 달라는 요청을 하는 복수의 애플리케이션 프로그램과,

상기 애플리케이션 프로그램이 상기 중앙 캐시를 판독 액세스하는 것을 허용하면서, 상기 관심 데이터를 상기 중앙 캐시에 추가하는, 상기 애플리케이션 프로그램과는 별개인 캐시 서비스 루틴을 포함하고,

상기 중앙 캐시에 상기 관심 데이터를 추가하는 동작은

첫째로, 상기 중앙 캐시에 상기 관심 데이터를 저장하고,

둘째로, 상기 관심 데이터에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 동작을 포함하고,

상기 관심 데이터에 대한 레퍼런스를 상기 중앙 캐시에 추가하는 동작은 원자적 동작(atomic operation)인 시스템.

명 세 서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

<23> 본 발명은 일반적으로 컴퓨터 메모리 저장 기술에 관한 것으로, 특히 캐시 메모리에 관한 것이다.

<24> 컴퓨터 애플리케이션이 필요로 하는 일부 데이터를 생성 또는 액세스하는 데에는 많은 비용이 든다. 이러한 비용에는 데이터를 계산하기 위한 계산 자원과, 네트워크 상에서 데이터를 액세스하기 위한 전송 비용(대역폭 및 시간 포함)이 포함될 수 있다. 컴퓨팅 장치는 종종, 일단 이들 데이터를 생성 또는 액세스하기 위해 자원을 소비한 후에는 "캐시" 메모리에 상기 데이터를 저장하게 된다. 그 후, 컴퓨팅 장치가 상기 데이터를 다시 필요로 하는 경우에는 캐시로부터 이들 데이터를 저렴한 비용으로 액세스할 수 있다.

<25> 캐시는 원래의 애플리케이션 또는 원래의 컴퓨팅 장치에 대하여 국지적(local)이거나, 또는 여러 애플리케이션 및 장치간에 공유될 수 있다. 후자와 같은 캐시의 유형은 종종 "중앙(central)" 캐시라 일컬어진다. 일부 환경에 있어서, 각 애플리케이션은 다른 애플리케이션과 중앙 캐시를 공유하면서도 자신만이 사용하기 위한 로컬 캐시(local cache)를 지원한다. 중앙 캐시는 하나 이상의 애플리케이션에 유용한 데이터를 저장하는 데 최적화되어 있지만, 로컬 캐시는 각 애플리케이션에 특유한 데이터를 캐싱(caching)하는 이점을 제공하는 데 쓰일 수 있다.

<26> 중앙 캐시에 있는 데이터의 관리는 간단하지 않다. 중앙 캐시로부터 데이터를 판독(read)하고자 하는 복수의 애플리케이션은 거의 문제를 야기하지 않지만, 적어도 하나의 애플리케이션이 상기 캐시에 데이터를 추가하고자 하는 경우에는 그러하지 못하다. 어떤 애플리케이션이 중앙 캐시에 기록(write)을 하는 것과 동시에 다른

애플리케이션이 상기 캐시로부터 데이터를 판독하는 것이 허용되는 경우, 판독하는 측은 오래된, 심지어는 왜곡된 데이터를 얻게 될 수 있다. 이러한 액세스 조정 문제는 하나 이상의 애플리케이션이 캐시에 데이터를 추가하고자 하는 경우에 더욱 악화된다.

<27> 이러한 액세스 조정 문제를 개선하는 일반적인 방식 중 하나는 "캐시 잠금(cache locking)"이라 불리는 방식이다. 어떤 애플리케이션이 캐시의 내용을 추가, 삭제, 또는 수정함으로써 이를 변경하고자 할 때마다, 상기 애플리케이션은 "록(lock)" 데이터 구조에 대한 단독 액세스를 시도한다. 록을 갖고 있는 동안 기록 애플리케이션은 캐시를 수정할 수 있으며, 다른 애플리케이션은 기록 애플리케이션이 록을 갖고 있는 한 캐시에 대한 액세스가 금지된다. 따라서 판독 애플리케이션이 오래된 또는 왜곡된 데이터를 얻는 것을 방지하게 된다. 두 애플리케이션 모두가 캐시를 수정하고자 하는 경우, 둘 중 하나는 다른 쪽이 록을 포기할 때까지 대기해야 한다.

발명이 이루고자 하는 기술적 과제

<28> 중앙 캐시에 대한 액세스를 조정함에 있어 록은 상당히 유용할 수 있다. 그러나 이로 인해, 어떤 애플리케이션이 캐시를 수정하고자 할 때마다 모든 애플리케이션의 액세스가 지연됨은 명백하다. 어떤 중앙 캐시의 경우에는, 애플리케이션이 이러한 속도 저하를 쉽게 견딜 수 있다. 그러나 다른 캐시의 경우에는 이것이 상당한 문제가 될 수도 있다. 예컨대, 폰트-글리프(font-glyph) 캐시의 경우를 살펴보자. 컴퓨터 스크린 상에 디스플레이되는 문자는 "글리프"라 불리는 개개의 요소들로 이루어진다. 이러한 글리프들 중 일부는 상당한 양의 데이터를 포함하고, 글리프들 중 일부는 그 생성에 있어 상당한 계산 자원을 소비하므로, 이들은 중앙 캐시를 적용하기에 이상적인 대상이다. 그러나 새로운 글리프를 중앙 캐시에 추가하면서 폰트-글리프 캐시를 잠그는 것은, 컴퓨터 스크린에 기록을 하는 애플리케이션에 있어 상당한 지연을 야기할 수 있다.

<29> 중앙 캐시가 이용할 수 있는 메모리 자원이 한정되어 있는 경우, 또 다른 캐시 관리 문제가 대두된다. 캐시에 데이터를 추가하고자 하는 복수의 애플리케이션은 서로 독립적으로 동작한다. 따라서 이들 애플리케이션 중 어떤 것도, 동작 환경을 전반적으로 향상시키기 위해서는 어떤 데이터가 중앙 캐시에 추가되어야 하는지에 관한 "포괄적(global)" 기준을 갖지 못한다. 중앙 캐시가 너무 커지게 되어, 후속 추가가 가능하도록 더 작은 크기로 재조직(reformulate)되는 경우에도 동일한 문제가 대두된다. 어떤 애플리케이션도 어느 데이터가 중앙 캐시에 보유되어야 할지, 그리고 장래의 캐시 증가를 대비하여 어느 데이터를 삭제하여 메모리를 비우는 것이 가장 나은지를 결정할 수 없다.

발명의 구성 및 작용

<30> 상기한 바에 비추어, 본 발명은 잠금으로 인한 지연 오버헤드(overhead) 없이 갱신(update)될 수 있고, 캐시 내 데이터의 중요도에 관한 포괄적 기준을 갖는 중앙 캐시를 제공한다. "원자적(atomic)" 갱신은 록에 의한 지연 오버헤드를 초래하지 않고 액세스 조정을 할 수 있는 이점을 제공한다. 캐시 갱신은 중도에 인터럽트(interrupt)될 수 없도록 설계되었다는 점에서 "원자적"이다. 그 결과, 캐시는 애플리케이션의 액세스 시에 항상 최신이자 일관된 상태가 된다.

<31> 애플리케이션은 항상 중앙 캐시의 데이터를 자유로이 판독할 수 있으며, 레퍼런스 테이블(reference table)을 통해 데이터를 액세스한다. 그러나 애플리케이션은 캐시를 직접 갱신하지 않으며, 대신 갱신 요청을 서비스 루틴으로 보낸다. 캐시를 갱신하기 위해, 캐시 서비스 루틴은 두 단계로 진행된다. 제1 단계에서, 캐시 서비스 루틴은 레퍼런스 테이블의 갱신 없이 새로운 데이터를 준비하여 이를 캐시에 추가한다. 이러한 제1 단계는 다소 시간이 걸릴 수 있으나, 캐시가 잠겨 있지 않으므로 캐시는 애플리케이션이 완전히 액세스할 수 있는 상태를 유지한다. 제1 단계 동안, 캐시를 액세스하는 애플리케이션은 새로운 데이터를 "보는 것"이 불가능한데, 이는 레퍼런스 테이블이 아직 갱신되지 않았기 때문이다. 캐시 데이터가 완전히 준비되어 캐시에 적재(load)되고 난 후에만 캐시 서비스 루틴은 갱신 프로세스의 제2 단계, 즉 레퍼런스 테이블의 갱신을 수행한다. 오직 하나의 포인터를 변경하는 것으로 이루어지는 이러한 갱신은 캐시를 잠그지 않은 상태에서 원자적으로 수행된다. 따라서 2단계 갱신 프로세스는 캐시를 잠그는 것을 필요로 하지 않으며, 액세스하는 애플리케이션에 대해 캐시가 항상 유효한 상태로 있도록 한다. 모든 갱신은 하나의 캐시 서비스 루틴에 의해 수행되기 때문에, 록들에 의해 복수의 캐시 기록 애플리케이션 사이를 조정할 필요가 없게 된다.

<32> 캐시 서비스 루틴은 캐시 내의 데이터가 어떻게 사용되는지에 관한 통계를 수집한다. 캐시가 너무 커지게 되면, 캐시 서비스 루틴은 이러한 통계를 사용하여 어느 데이터가 새로운 캐시로 복사되어야 할지를 결정한다. 새로운 캐시는 원자적으로, 그리고 두 단계로 생성된다. 제1 단계 동안 캐시 서비스 루틴은 새로운 캐시를 생

성하며, 여기에 이전의 캐시로부터 선택된 데이터를 배치한다. 애플리케이션은 아직 새로운 캐시에 대해서 알지 못한다. 새로운 캐시를 이용할 준비가 되면, 캐시 서비스 루틴은 그 새로운 캐시에 대한 레퍼런스(reference)를 이전 캐시의 헤더에 추가한다. 그 후 제2 단계에서는 또 다른 원자적 동작(operation)을 이용하여, 캐시 서비스 루틴은 이전의 캐시를 "폐기(obsolete)"로 표지(mark)한다. 이전의 캐시가 폐기된 것으로 표지되었음을 알게 되면, 애플리케이션은 새로운 캐시에 대한 레퍼런스를 따라가 새로운 캐시만을 사용하기 시작한다. 하나의 캐시 내에서의 갱신과 마찬가지로, 전체 캐시를 교체하기 위한 이러한 메커니즘은 애플리케이션이 항상 일관된 캐시를 보도록 수행된다.

<33> 애플리케이션은 폐기 플래그(flag)를 인식하여 새로운 캐시로 전환할 때까지는 폐기된 캐시를 계속 사용할 수 있다. 일단 모든 애플리케이션이 전환되면, 폐기된 캐시는 자동으로 삭제된다.

<34> 일부 실시예에서, 캐시 내의 레퍼런스 테이블은 캐시에 저장된 데이터의 위치를 캐시 내의 다른 위치에 대해 상대적으로 특정하는 오프셋(offset)을 포함한다. 이로 인해, 캐시를 파일로서 저장할 수 있으며, 캐시를 호스트(host)하는 컴퓨팅 장치가 재부팅된 후에도 즉시 캐시를 다시 사용할 수 있다는 장점이 생긴다.

<35> 중앙 캐시는 하나의 컴퓨팅 장치에 의해 호스트될 수 있으며, 상기 컴퓨팅 장치 및 다른 컴퓨팅 장치에 있는 애플리케이션에 의해 이용될 수 있다. 각 애플리케이션은 또한 중앙 캐시와 함께 사용할 자신만의 로컬 캐시를 가질 수도 있다. 로컬 캐시가 중앙 캐시와 동일한 데이터 구조를 갖는 경우, 이들 두 캐시에 대해 동일한 캐시 액세스 코드를 사용할 수 있다.

<36> 캐시 서비스 루틴은, 캐시를 교체할 경우에 어떤 데이터를 보존할지를 결정하기 위해 수집한 캐시 사용에 관한 통계에 휴리스틱(heuristic)을 적용한다. 일부 실시예에서는, 휴리스틱을 변경하고 캐시의 동작을 감시할 수 있도록 하는 사용자 인터페이스가 제공된다.

<37> 첨부된 청구항이 본 발명의 특징을 면밀히 나타내고 있으나, 첨부된 도면 및 이하의 상세한 설명으로부터 본 발명을 그 목적 및 효과와 함께 가장 잘 이해할 수 있을 것이다.

<38> 도면에 있어서 동일한 참조 번호는 동일한 요소를 나타내는 것이며, 도면은 본 발명을 적합한 컴퓨팅 환경에 구현한 것을 도시하고 있다. 이하의 설명은 본 발명의 실시예에 기초한 것으로, 본 명세서에서 명시적으로 기재되지 않은 대체 실시예와 관련하여 본 발명을 한정하는 것으로 받아들여서는 아니된다.

<39> 이하의 설명에 있어서, 달리 표시하지 않은 경우 하나 또는 그 이상의 컴퓨팅 장치에 의해 수행되는 동작의 행위 및 기호적 표현을 참조하여 본 발명을 설명하였다. 마찬가지로, 컴퓨터로 실행된다고 종종 일컬어지는 이러한 행위 및 동작은 구조화된 형식의 데이터를 나타내는 전자적 신호를 컴퓨팅 장치의 처리 유닛에 의해 조작하는 것을 포함한다. 이러한 조작을 통해 데이터를 변형시키거나 데이터를 컴퓨팅 장치의 메모리 시스템 내의 여러 위치에 유지하는데, 이는 본 기술 분야의 당업자가 용이하게 이해할 수 있는 방식으로 상기 장치의 동작을 재구성 또는 변경하게 된다. 데이터가 유지되는 데이터 구조들은, 데이터의 형식에 의해 정의되는 특정한 속성을 갖는 물리적 메모리 위치들이다. 비록 본 발명을 이상과 같은 맥락에서 설명하고 있지만, 이하에 설명할 다양한 행위 및 동작은 또한 하드웨어에서도 구현될 수 있음을 본 기술 분야의 당업자가 이해할 수 있듯이 본 발명은 이에 한정되는 것이 아니다.

<40> 본 발명은 원자적으로 갱신되는 중앙 캐시 메모리를 제공한다. 중앙 캐시는 캐시를 호스트하는 컴퓨팅 장치에서 실행되는 애플리케이션에 의해 독점적으로 사용되거나, 또는 도 1a에 나타낸 바처럼 여러 컴퓨팅 장치의 애플리케이션에 의해 사용될 수도 있다. 도 1a에는 캐시 데이터 공유 환경(cached-data-sharing environment)(100)에 있는 세 컴퓨팅 장치인 A(102), B(104) 및 랩톱(106)이 도시되어 있다. 중앙 캐시(도시되지 않음)는 컴퓨팅 장치 A(102)에 존재하며, 애플리케이션은 LAN(108)을 통해 중앙 캐시에 액세스할 수 있다. 공유 환경(100)의 컴퓨팅 장치 사이에서 캐시 요청 및 응답을 전송하기 위한 표준 통신 프로토콜이 존재한다.

<41> 인터넷(110)에 대한 접속이 도시되어 있는데, 이는 원격 컴퓨팅 장치라도 캐시 데이터 공유 환경(100)에 참여할 수 있음을 나타내는 것이다. 실제로는, 이러한 원격 장치가 중앙 캐시를 액세스하는 데에는 긴 통신 시간이 필요하게 되는데, 이는 빠른 데이터 액세스를 제공한다는 캐시의 목표에 반하는 것이다. 대부분의 중앙 캐시 시나리오는 오직 하나, 또는 많아야 몇 개의 가까이 위치한 컴퓨팅 장치에 수반되는 것이다.

<42> 도 1b는 도 1a에 나타낸 캐시 데이터 공유 환경(100)의 실시예에 관한 구조적 세부 사항을 나타낸 것이다. 애플리케이션 A(112)는 컴퓨팅 장치 A(102)에서 실행된다. 애플리케이션 A(112)를 함께 구성하는 루틴들 중에는 로컬 및 중앙 캐시 검색 루틴(114)이 있다. 애플리케이션 A(112)가 캐시 내에 위치할 수 있는 데이터를 필요로 하는 경우, 이러한 루틴(114)은 현재의 로컬 캐시(116) 및 현재의 중앙 캐시(118)에 있는 데이터를 찾는다. 로

컬 캐시(116)는 애플리케이션 A(112)의 일부이자 그 제어 하에 있다. 중앙 캐시(118)는 애플리케이션 A(112)의 일부가 아니지만, 애플리케이션 A(112) 및 다른 애플리케이션은 이를 액세스할 수 있다.

<43> 중앙 캐시 검색 루틴(114)이 요청된 데이터를 로컬 캐시(116) 또는 중앙 캐시(118)에서 찾아 낸 경우, 그 데이터를 애플리케이션 A(112)에 반환한다. 그렇지 않은 경우에는 데이터를 다른 장소에서 찾거나, 또는 데이터를 생성한다. 일단 데이터를 찾거나 생성하면, 캐시 검색 루틴(114)은 로컬 캐시 갱신 루틴(120)을 호출함으로써 상기 데이터를 로컬 캐시(116)에 추가시킬 것을 요청한다. 상기 데이터를 중앙 캐시(118)에 추가시키라는 요청은 중앙 캐시 서비스 루틴(124)에도 전달된다.

<44> 데이터가 추가될 경우 로컬 및 중앙 캐시 모두는 증가하게 된다. 이들이 너무 커지게 되면, 새로운 캐시를 생성하여 이전 캐시의 데이터 중 일부를 배치한다. 어느 데이터가 새로운 캐시로 전달될지를 선택하기 위해 임의의 개수의 방법이 적용된다. 예컨대 가장 최근에 사용된 데이터가 선택되거나, 또는 가장 자주 사용되는 데이터가 선택된다. 도 1b는 현재의 로컬 캐시(116)와 함께 이전의 로컬 캐시(122)를 나타내고 있다.

<45> 새로운 중앙 캐시를 생성하는 경우, 중앙 캐시를 사용하는 애플리케이션의 중단(disruption)을 방지하기 위한 조치를 취하게 된다. 새로운 캐시(118)가 준비되면, 이전의 중앙 캐시(126)를 "폐기"로 표지하여 애플리케이션에게 새로운 캐시를 이용할 수 있음을 알린다. 그러나 이들 애플리케이션은 즉시 새로운 캐시(118)로 전환할 필요는 없으며, 당분간 이전의 중앙 캐시(126)를 계속 액세스하는 것을 선택할 수 있다. 애플리케이션들이 더 이상 이전의 중앙 캐시(126)를 액세스하지 않게 되면, 그 캐시는 삭제된다.

<46> 애플리케이션 B(128)는 캐시 데이터 공유 환경(100)에 있는 다른 컴퓨팅 장치에서 실행된다. 이 애플리케이션은 로컬 캐시가 없는 것으로 도시되어 있지만, 중앙 캐시 검색 루틴(130)을 갖고 있다. 이 루틴은 아직 이 캐시가 폐기된 것으로 표지되었음을 알지 못하기 때문에 여전히 이전의 중앙 캐시(126)를 액세스한다. 캐시 검색 루틴(130)은 중앙 캐시에 데이터를 추가할 것을 요청할 수 있지만, 중앙 캐시 서비스 루틴(124)은 그 데이터를 폐기된 캐시(126)가 아닌 현재의 중앙 캐시(118)에 추가할 것이다.

<47> 도 1의 컴퓨팅 장치(102, 104 및 106)들은 임의의 아키텍처를 가질 수 있다. 도 2는 본 발명을 지원하는 컴퓨터 시스템의 예를 일반적으로 나타낸 블록도이다. 도 2의 컴퓨터 시스템은 적합한 환경의 한 가지 예일 뿐이며, 본 발명의 용도나 기능의 범위를 한정하고자 하는 것이 아니다. 컴퓨팅 장치(102)는 도 2에 나타낸 구성요소 중 하나 또는 이들의 조합에 대해 의존 관계 또는 필요 조건을 갖는 것으로 해석되어서는 아니 될 것이다. 본 발명은 다른 많은 범용 또는 특수 용도 컴퓨팅 환경 또는 구성과 함께 동작한다. 본 발명과 함께 쓰이기에 적합한, 잘 알려진 컴퓨팅 시스템, 환경 및 구성의 예에는 개인용 컴퓨터, 서버, 핸드헬드(hand-held) 또는 랩톱 장치, 태블릿(tablet) 장치, 멀티프로세서 시스템, 마이크로프로세서 기반의 시스템, 셋톱 박스, 프로그램 가능한 소비자 가전 기기, 네트워크 PC, 미니컴퓨터, 메인프레임 컴퓨터, 그리고 상기 시스템 또는 장치 중 임의의 것을 포함하는 분산형 컴퓨팅 환경이 포함되지만, 이에 한정되는 것은 아니다. 가장 기본적인 구성에 있어서, 컴퓨팅 장치(102)는 통상적으로 적어도 하나의 처리 유닛(200)과 메모리(202)를 포함한다. 메모리(202)는 휘발성(예컨대 RAM), 비휘발성(예컨대 ROM 또는 플래시 메모리), 또는 이들의 어떠한 조합일 수 있다. 이러한 가장 기본적인 구성은 도 2에 점선(204)으로 도시되어 있다. 컴퓨팅 장치(102)는 추가적인 특징 및 기능을 가질 수 있다. 예컨대 컴퓨팅 장치(102)는 추가적인 저장 장치(이동형 및 고정형)를 포함할 수 있는데, 여기에는 자기 및 광 디스크 및 테이프가 포함되지만 이에 한정되는 것은 아니다. 이러한 추가적인 저장 장치는 도 2에서 이동형 저장 장치(206) 및 고정형 저장 장치(208)로 도시되어 있다. 컴퓨터 저장 매체에는 휘발성 및 비휘발성, 이동형 및 고정형 저장 매체는 물론, 컴퓨터로 판독 가능한 명령어, 데이터 구조, 프로그램 모듈, 또는 기타 데이터 등의 정보를 저장하는 임의의 방법 또는 기술로 구현된 매체가 포함된다. 메모리(202), 이동형 저장 장치(206) 및 고정형 저장 장치(208)는 모두 컴퓨터 저장 매체의 예이다. 컴퓨터 저장 매체에는 RAM, ROM, EEPROM, 플래시 메모리, 기타 메모리 기술, CD-ROM, DVD, 기타 광 저장 장치, 자기 카세트, 자기 테이프, 자기 디스크 저장 장치, 기타 자기 저장 장치, 그리고 원하는 정보를 저장하는 데 쓰일 수 있고 장치(102)가 액세스할 수 있는 기타 임의의 매체가 포함되지만, 이에 한정되는 것은 아니다. 임의의 이러한 컴퓨터 저장 매체는 장치(102)의 일부가 될 수도 있다. 장치(102)는 또한 다른 장치와 통신할 수 있도록 해 주는 통신 채널(210)을 포함할 수 있다. 통신 채널(210)은 통신 매체의 예이다. 통상적으로 통신 매체는 컴퓨터로 판독 가능한 명령어, 데이터 구조, 프로그램 모듈, 또는 반송파(carrier wave)나 그 밖의 전송 메커니즘 등의 변조 데이터 신호 내의 데이터를 구현한 것이며, 임의의 정보 전달 매체를 포함한다. "변조 데이터 신호(modulated data signal)"라는 용어는, 신호 내의 정보를 인코딩하는 등의 방식으로 신호의 하나 또는 그 이상의 특성이 설정 또는 변경된 신호를 말한다. 예컨대 통신 매체에는 유선 매체(예컨대 유선 네트워크 및 직접 유선 접속) 및 무선 매체(예컨대 음향, RF, 적외선 및 기타 무선 매체)가 포함되나, 이에 한정되는 것은 아니다. 본 명세서에서 쓰

인 "컴퓨터로 판독 가능한 매체"라는 용어는 저장 매체 및 통신 매체 모두를 포함한다. 컴퓨팅 장치(102)는 또한 키보드, 마우스, 펜, 음성 입력 장치, 태블릿, 접촉(touch) 입력 장치 등과 같은 입력 장치(212)를 가질 수 있다. 디스플레이(접촉 입력 장치와 통합될 수 있음), 스피커 및 프린터와 같은 출력 장치(214)가 또한 포함될 수 있다. 이러한 모든 장치는 본 기술 분야에 주지되어 있으며 본 명세서에서 상세히 논할 필요가 없다.

<48> 도 3a 내지 도 3d는 도 1b의 로컬 및 중앙 캐시 검색 루틴(114)을 위한 방법의 예를 나타내고 있다. 이 흐름도는 캐시 검색 루틴(114)의 모든 실시예에 포함될 필요가 있는 것은 아닌 많은 옵션을 포함한다.

<49> 흐름도가 도 3a의 단계(300)에서 시작되기에 앞서, 애플리케이션 A(112)의 관심 대상인 데이터를 액세스하기 위한 요청이 캐시 검색 루틴(114)에게 이루어진다. 단계(300)에서, 중앙 캐시의 액세스 가능 여부를 체크함으로써 캐시 검색 루틴(114)이 개시된다. 중앙 캐시를 액세스할 수 없는 경우, 캐시 검색 루틴(114)은 도 3d의 단계(326)로 진행하여 상기 관심 데이터(data of interest)를 로컬 캐시에서 찾는다. 상황에 따라, 다른 실시예의 캐시 검색 루틴(114)은 중앙 캐시에 앞서 로컬 캐시에 대한 액세스를 시도하기도 한다.

<50> 중앙 캐시를 액세스할 수 있는 경우, 단계(302)에서 캐시 검색 루틴(114)은 중앙 캐시가 "폐기"된 것으로 표지되었는지를 체크한다. 여기서 이 단계 및 이후의 단계(304)는 선택적인 것임에 주목하자. 즉 캐시 검색 루틴(114)은 폐기된 캐시를 계속 사용할 수 있다. 그러나 최신 상태를 유지하기 위해서, 캐시 검색 루틴(114)은 액세스중인 중앙 캐시의 상태를 주기적으로 체크해야 하며, 이전의 캐시가 폐기되었음을 알게 된 경우에는 이전의 캐시를 해제(release)해야 한다.

<51> 중앙 캐시를 폐기된 것으로 표지하기에 앞서, 중앙 캐시 서비스 루틴(124)은 새로운 중앙 캐시를 생성시킨다(도 5a 및 도 5b의 단계 512 내지 518 참조). 따라서 단계(304)에서 캐시 검색 루틴(114)은 폐기된 중앙 캐시의 헤더에 있는, 현재의 중앙 캐시에 대한 레퍼런스를 따라갈 수 있다. 요청이 도착하는 시점과 그 요청이 처리되는 시점 사이에는 상당한 지연이 존재할 수 있으므로, 이러한 지연 도중에 하나 이상의 캐시가 폐기된 것으로 표지될 수 있다. 이러한 경우 단계(302 및 304)는 폐기되지 않은 캐시에 도달할 때까지 반복된다.

<52> 단계(306)에서 캐시 검색 루틴(114)은 중앙 캐시를 액세스하여 이것이 관심 데이터를 포함하는지 여부를 살펴본다. 단계(306)가 어떤 식으로 수행되는지에 대한 상세한 사항은 중앙 캐시가 어떠한 구조를 가졌는지에 따라 크게 달라질 수 있다. 도 3a 내지 도 3c에 나타낸 단계(306)는 한 가지 가능한 구현예를 나타낸 것이며, 도 4a 및 도 4b에 나타낸 캐시 데이터 구조의 예와 함께 고려되어야 하는 것이다.

<53> 도 4a는 단계(306)에 도시한 프로시저(procedure)와 함께 쓰일 수 있는 중앙 캐시(400)의 구조를 나타내고 있다. 고정된 크기를 갖는 헤더(402)는 폐기 플래그를 포함하며, 이 플래그가 설정된 경우에는 더욱 최신의 중앙 캐시(400)에 대한 레퍼런스를 포함한다. 다음은 고정된 크기를 갖는 레퍼런스 테이블(404)이다. 레퍼런스 테이블(404)의 각 항목은 데이터 요소 저장 영역(406)에 있는 데이터 요소(도 4b의 410 참조)에 대한 레퍼런스, 또는 상기 항목이 데이터 요소를 참조하지 않음을 나타내는 특별한 값(예컨대 NULL)이다. 일부 실시예에서는, 이러한 레퍼런스가 정수 오프셋으로 주어진다. 이에 의해 중앙 캐시(400)는 디스크에 파일로서 저장될 수 있으며, 호스트 컴퓨팅 장치(102)가 재부팅된 후에 복구될 수 있다.

<54> 실제 캐시 데이터는 개별 데이터 요소(410)로서 저장된다. 이들 데이터 요소(406)에 할당된 공간은 필요에 따라 유휴 공간(408)으로 확장된다. 유휴 공간(408)을 모두 사용하게 되는 시점이, 캐시 서비스 루틴(124)이 새로운 중앙 캐시(400)를 생성해야 하는 시점이다.

<55> 단계(306)에 나타낸 특정한 구현예는 단계(308)에서 관심 데이터를 해싱(hashing)함으로써 개시된다. 그 후 단계(310)에서 해시는 중앙 캐시(400)의 레퍼런스 테이블(404)에 대한 색인으로서 사용된다. 도 3b의 단계(312)에서는 레퍼런스 테이블(404)의 선택된 항목을 검사한다. 이 항목이 어떤 데이터 요소도 참조하지 않는 경우, 관심 데이터는 중앙 캐시(400)에 존재하지 않는다(단계 314). 다음으로 캐시 검색 루틴(114)은 도 3d의 단계(326)로 진행하여 중앙 캐시(400) 이외의 장소에서 관심 데이터를 찾는다.

<56> 반면 레퍼런스 테이블(404)의 선택된 항목이 데이터 요소를 참조하는 경우, 그 데이터 요소는 관심 데이터를 포함하거나 포함하지 않을 수 있다. 그 이유는, 상이한 데이터 값이 동일한 값으로 해싱되어 레퍼런스 테이블(404)에 있는 동일한 항목을 선택하게 될 수도 있기 때문이다.

<57> 도 4b의 데이터 요소 구조(410)의 예를 살펴보자. 데이터 요소(410) 각각은, 동일한 해시값을 갖는 다른 데이터 요소(410) 또는 상기 항목이 다른 데이터 요소(410)를 참조하지 않음을 나타내는 특별한 값(예컨대 NULL)을 참조하는 레퍼런스(412)로 시작한다. 레퍼런스 테이블(404)에 있는 항목들과 마찬가지로, 이러한 레퍼런스는 정수 오프셋으로 주어질 수 있다. 레퍼런스(412) 다음에는 이 데이터 요소(410)에 포함된 데이터를 고유하게

식별하는 데이터 식별자 필드(414)가 있다. 마지막으로 데이터 자체를 포함하는 필드(416)가 있다. 이 필드(416)의 크기 및 구조는 저장된 데이터의 성질에 특유한 것이다. 어떤 실시예에서, 데이터 요소(410)는 자신의 크기를 나타내는 필드, 또는 그에 상당하는, 요소 특유(element-specific)의 데이터 필드(416)의 종단 위치를 나타내는 필드를 포함한다. 다른 실시예에서, 중앙 캐시(400)의 헤더(402)에 있는 필드는 할당된 데이터 저장 영역(406)의 종단을 가리킨다. 어느 경우라도, 이러한 길이 필드는 캐시(400)의 어느 위치에 또 다른 데이터 요소(410)가 추가될 수 있는지를 나타내는 데에 사용된다(이하 설명할 도 5d의 단계 534).

<58> 데이터 요소 구조(410)의 이러한 예를 염두에 두고, 도 3b의 단계(316)로 돌아가 보자. 단계(316)에서 데이터 요소(410)의 해시는 관심 데이터의 해시와 일치한다. 이 데이터 요소(410)가 관심 데이터를 포함하는지를 살펴보기 위해, 데이터 요소(410)의 데이터 식별자 필드(414)를 관심 데이터와 비교한다. 이들이 일치하는 경우 검색이 완료된다. 관심 데이터는 단계(318)에서 인출(retrieve)되어 애플리케이션 A(112)로 전달된다. 이에 의해 도 3a 내지 도 3d의 캐시 검색 루틴(114)이 성공적으로 종료된다.

<59> 반면, 단계(316)에서의 비교 결과 이러한 데이터 요소(410)가 관심 데이터와 일치하지 않는 경우, 캐시 검색 루틴(114)은 도 3c의 단계(320)로 진행한다. 단계(320)에서, 데이터 요소(410)의 레퍼런스 필드(412)를 검사한다. 이 필드가 다른 데이터 요소(410)를 참조하지 않는 경우, 중앙 캐시(400)는 관심 데이터를 포함하지 않는다(단계 322). 캐시 검색 루틴(114)은 도 3d의 단계(326)로 진행하여 관심 데이터를 다른 위치에서 검색한다.

<60> 단계(320)에서, 데이터 요소(410)의 레퍼런스 필드(412)가 다른 데이터 요소를 참조하는 경우, 그 다른 데이터 요소를 검사하여 그것이 관심 데이터를 포함하는지 여부를 살펴본다. 단계(324)는 이하의 프로세스를 나타낸다. 즉 관심 데이터를 발견하여 인출할 때까지(단계 318) 또는 관심 데이터를 발견하지 못하고 데이터 요소(410) 연쇄(chain)의 종단에 이를 때까지 단계(316) 내지 단계(322)를 반복함으로써 데이터 요소(410)의 연쇄(chain)를 추적한다.

<61> 중앙 캐시(400)가 관심 데이터를 포함하지 않는 경우, 도 3d의 단계(326)에서 캐시 검색 루틴(114)은 애플리케이션 A(112)의 로컬 캐시(116)를 검색할 수 있다. 일부 구현예에 있어서, 단계(326)의 세부 사항은 단계(306)의 세부 사항과 유사하다. 이에 의해 캐시 검색 코드를 재활용할 수 있게 된다. 어느 경우에 있어서도, 로컬 캐시(116)가 관심 데이터를 포함하는 경우에는 단계(322)에서 이들 데이터가 인출된다. 그렇지 않다면, 그리고 관심 데이터를 검색할 다른 캐시가 존재하지 않는다고 가정하면, 단계(328)에서 이들 데이터가 생성된다. 그 후 이들 데이터는 추후의 액세스를 촉진하도록 단계(330)에서 로컬 캐시(116)에 추가될 수 있다. 단계(334)에서는 생성된 상기 데이터를 중앙 캐시(400)에 추가시켜달라는 요청을 할 수 있다.

<62> 도 3a 내지 도 3d는 캐시 검색 루틴(114)이 중앙 캐시(400) 또는 다른 장소로부터 관심 데이터를 어떻게 인출하는지를 나타내고 있다. 도 5a 내지 도 5d는 캐시(400)의 또 다른 태양, 즉 데이터를 캐시(400)에 추가시키고 현재의 캐시가 가득 차게 되면 새로운 중앙 캐시를 생성시키는 캐시 서비스 루틴(124)을 나타내고 있다. 캐시 서비스 루틴(124)은 캐시(400)에 데이터를 추가시키는 유일한 루틴이므로, 복수의 기록 애플리케이션 사이의 충돌을 방지하도록 캐시(400)를 잠글 필요가 없다. 또한 모든 기록 요청은 캐시 서비스 루틴(124)을 통과하므로, 캐시 서비스 루틴(124)은 캐시 사용에 관한 통계를 수집하여 캐시(400)에 데이터를 추가하라는 특정한 요청의 중요도에 관한 "포괄적"인 기준을 형성할 수 있으며, 새로운 캐시가 생성되는 경우 어느 데이터를 가지고 가야 할지 결정할 수 있다. 도 5a 내지 도 5d의 프로시저는 예시적인 것일 뿐이다. 이는 도 4a 내지 도 4b에서 소개된 중앙 캐시 데이터 구조를 사용한다.

<63> 캐시 서비스 루틴(124)은 도 5a의 단계(500)에서 중앙 캐시(400)에 대한 데이터 추가 요청을 수신하였을 때에 개시된다. 이러한 요청은 캐시 검색 루틴(114)이 도 3d의 단계(334)를 수행하였을 때에 기인한 것일 수 있다.

<64> 단계(502)에서, 일부 실시예의 캐시 서비스 루틴(124)은 수신된 요청 및 다른 요청에 관한 통계를 수집한다. 단계(504)에서, 캐시 서비스 루틴(124)은 이러한 요청에 응할 것인지를 결정한다. 일부 실시예에서는 모든 요청에 단순히 응하지만, 다른 실시예에서는 관심 데이터가 중앙 캐시(400)에 추가될 "가치"가 있는지를 결정하기에 앞서 수집된 통계를 참조한다. 일부 실시예에서는 관심 데이터의 크기가 캐시(400)의 유휴 공간(408)과 비교했을 때 지나치게 큰 경우에는 이러한 요청을 거부한다. 관심 데이터를 캐시(400)의 데이터와 비교하고(도 3a 내지 도 3c의 단계 306을 수행함으로써 이루어질 수 있음), 관심 데이터가 이미 존재하는 경우에는 이러한 요청을 거부한다. 이러한 상황은, 단계(500)에서 수신된 요청을 대기열(queue)에 두어, 요청이 도달한 시점과 관심 데이터가 캐시에 추가되는 시점 사이에는 상당한 지연이 생길 수 있기 때문이다(도 5a 내지 도 5d의 단계 522). 지연되는 동안, 요청을 하는 캐시 서비스 루틴(114) 또는 다른 캐시 서비스 루틴은 캐시(400)를 다시 액

세스하고, 관심 데이터를 발견하지 못하여 다시 요청을 제출할 수 있다. 나중의 요청이 처리될 때까지는, 이전의 요청에 응하여 관심 데이터가 캐시(400)에 이미 추가된 상태이다. 어느 경우라도, 단계(506)에서 상기 요청에 응하지 않는다는 결정에 도달한 경우, 캐시 서비스 루틴(124)은 후속 요청을 기다리게 된다(단계 508).

<65> 상기 요청에 응한다는 결정을 한 경우에는, 단계(510)에서 캐시 서비스 루틴(124)은 새로운 중앙 캐시(400)가 필요한지 여부를 결정한다. 예컨대, 관심 데이터가 캐시(400)의 유휴 공간(408)에 들어맞지 않을 경우에는, 도 5a 및 도 5b의 단계(512) 내지 단계(520)에서 새로운 캐시(400)가 생성된다. 레퍼런스 테이블(404)이 가득 차게 되거나, 또는 오랫동안 사용되지 않았던 데이터 요소(410)를 캐시(400)가 지나치게 많이 갖고 있는 경우에는 새로운 캐시(400)가 생성될 수도 있다. 새로운 캐시(400)가 필요하지 않은 경우, 캐시 서비스 루틴(124)은 곧바로 도 5b의 단계(522)에서 관심 데이터의 추가를 진행한다.

<66> 새로운 중앙 캐시(400)가 필요한 경우에는, 비어 있는 쉘(shell)이 단계(512)에서 생성된다. 캐시는 일반적으로 RAM에 저장되어 이들의 데이터에 대한 빠른 액세스를 제공할 수 있도록 한다. 일부 운영 체제는 파일 시스템이 RAM의 영역을 매핑(mapping)할 수 있도록 한다. 이는 이후 설명할 소정의 장점을 제공하게 된다.

<67> 도 5b의 단계(514)에서, 캐시 서비스 루틴(124)은 기존의 캐시로부터 선택된 데이터 요소(410)를 새로이 생성된 캐시(400)에 배치시킨다. 데이터 요소(410)가 선택되는 방식에는 여러 가지가 있을 수 있다. 캐시 서비스 루틴(124)은 데이터 요소(410)가 어떻게 사용되었는지에 관한 통계를 수집할 수 있다. 이어서, 가장 최근에 사용된 요소 또는 가장 자주 쓰인 요소가 선택된다. 일부 실시예에서는 무작위로 선택이 이루어진다. 어느 경우라도, 일단 데이터 요소(410)가 선택되면, 이들은 새로운 중앙 캐시(400)의 데이터 저장 영역(406)으로 복사되고, 새로운 캐시(400)의 레퍼런스 테이블(404)이 채워진다. 단계(514)를 실행하는 데는 다소 시간이 걸리지만, 이 단계동안 이전의 캐시에 항상 액세스할 수 있음에 주목하자.

<68> 단계(516)에서는 새로운 캐시(400)에 대한 레퍼런스가 이전 캐시(402)의 헤더에 기록되고, 단계(518)에서는 이전의 캐시를 "폐기"된 것으로 표지하는 플래그가 설정된다. 이제 새로운 캐시(400)를 이용할 수 있는 준비가 되었다. 캐시 검색 루틴(114)이 이전의 캐시에서 폐기 플래그를 보게 되면, 새로운 캐시(400)에 대한 레퍼런스를 따라간다(도 3a의 단계 302 및 304 참조).

<69> 단계(520)에서 캐시 서비스 루틴(124)은, 폐기된 캐시를 참조하는 애플리케이션이 더 이상 존재하지 않게 되면, 즉 모든 애플리케이션이 폐기된 캐시에 대한 레퍼런스를 해제하게 되면, 곧바로 이를 자동으로 삭제할 것을 운영 체제에 요청한다.

<70> 새로운 캐시(400)가 막 생성되었는지 여부에 관계없이, 관심 데이터는 단계(522)에서 현재의 캐시(400)에 추가된다. 도 5b 내지 도 5d의 단계(522)는 도 3a 내지 도 3c의 단계(306)와 매우 유사한데, 이는 단계(522)에서는 단계(306)에서 검색되는 데이터 구조들을 생성하기 때문이다. 도면에서 단계(306) 및 단계(522)에 대한 상세한 특징은 오직 일부의 실시예를 설명하기 위한 것이지만, 대부분의 실시예에 있어서 이들 단계는 서로 매우 유사하다.

<71> 단계(522)의 특정한 실시예는 단계(524)(도 3a의 단계 308과 유사)에서 관심 데이터를 해싱하는 것으로 개시된다. 단계(526)에서, 해시를 사용하여 캐시(400)의 레퍼런스 테이블(404)에 있는 항목을 선택한다(단계 310과 유사). 단계(528)에서, 선택된 상기 항목이 아직 데이터 요소(410)를 참조하고 있지 않는 경우, 데이터 요소(410)는 도 5d의 단계(534)에서 캐시(400)의 데이터 저장 영역(406)에 생성된다. 그 후 이 새로운 데이터 요소에 관심 데이터가 배치된다. 이러한 배치 단계가 진행되는 동안, 캐시(400)를 액세스하는 캐시 검색 루틴(114)은 새로운 데이터를 보지 못하고, 따라서 이들과 동일한 데이터를 추가시켜달라는 추가적인 요청을 할 수도 있다. 일단 배치 단계(534)가 완료되고 새로운 데이터 요소(410)가 준비되면, 캐시 서비스 루틴(124)은 레퍼런스 테이블(404)의 선택된 포인터가 새로운 데이터 요소(410)를 참조하도록 상기 선택된 포인터를 원자적으로 갱신한다(단계 536). 레퍼런스를 기록하는 것은 하나의 컴퓨터 명령어만을 차지하므로, 본래적으로 인터럽트를 야기하는 성질의 것이 아니다. 따라서 내부 일관성의 유지를 위해 이러한 동작 도중에 캐시(400)를 잠글 필요가 없다.

<72> 단계(528)로 돌아가서, 레퍼런스 테이블(404)의 선택된 항목이 이미 데이터 요소(410)를 참조하는 경우, 단계(530)에서 데이터 요소(410)의 레퍼런스 필드(412)를 검사한다. 단계(532)에서, 레퍼런스의 연쇄(chain)가 그 종단에 이를 때까지 추적된다(이는 도 3b 및 도 3c의 단계 316 내지 단계 324에서 동일한 레퍼런스 연쇄를 검색하는 것과 매우 유사함). 일단 레퍼런스 연쇄(chain)의 종단에 도달하면, 캐시 서비스 루틴은 앞서 설명한 바처럼 단계(534)로 진행하고, 이 단계에서 새로운 데이터 요소(410)가 할당되어 관심 데이터로 채워진다. 이러

한 경우, 단계(536)는 상기 레퍼런스 연쇄(chain)의 종단에 있던 기존의 데이터 요소(410)에 새로운 데이터 요소(410)에 대한 레퍼런스를 원자적으로 추가하여, 상기 레퍼런스 연쇄(chain)를 하나의 "링크"만큼 확장시킨다.

<73> 요컨대, 새로운 데이터 요소(410)를 기존의 중앙 캐시(400)에 추가하건, 또는 새로운 중앙 캐시(400)를 생성하건, 캐시 서비스 루틴(124)은 두 단계로 진행된다. 우선, 데이터가 준비되고, 시간을 소모하는 모든 작업이 완료된다. 이 단계 동안, 캐시 검색 루틴(114)은 중앙 캐시(400)에 대한 변경 사항을 알지 못한다. 따라서 내부 일관성을 유지하기 위해 중앙 캐시(400)를 잠글 필요가 없다. 또한 캐시 서비스 루틴(124)이 높은 우선권을 가지고 실행될 필요가 없다. 제2 단계는 새로운 데이터 요소(410) 또는 새로운 중앙 캐시(400)에 하나의 포인터를 기록하는 단계를 포함한다. 단일 포인터를 기록하는 것은 본래 인터럽트를 야기하지 않는 프로시저이므로, 마찬가지로 이 단계 동안 중앙 캐시(400)를 잠글 필요가 없다. 일단 제2 단계가 완료되면, 캐시 검색 루틴(114)은 새로운 데이터 또는 새로운 캐시를 액세스할 수 있다.

<74> 도 6은 캐시 서비스 루틴(124)의 동작을 감시 및 구성하는 방법을 나타내고 있다. 단계(600)에서는, 중앙 캐시의 최대 및 현재 크기가 디스플레이된다. 단계(602)에서는, 새로운 중앙 캐시에 데이터 요소를 배치할 때 기존의 중앙 캐시로부터 데이터 요소를 선택하기 위해 도 5b의 단계(514)에서 사용되는 휴리스틱이 디스플레이된다. 중앙 캐시에 대한 데이터 추가 요청에 응할 것인지 여부를 결정하기 위해 단계(504)에서 사용되는 휴리스틱이 단계(604)에서 디스플레이된다. 이러한 휴리스틱은 단지 "항상 요청에 응함"일 수 있음을 상기하자. 단계(606)에서는 캐시 사용에 관하여 수집된 여러 통계가 디스플레이된다. 관리자는 이러한 통계를 분석하여 중앙 캐시가 최적으로 동작하는지 않는다고 결정할 수 있다. 관리자는 단계(610)에서 반영되는 일부 변경 사항을 단계(608)에서 입력하며, 이는 단계(610)에서 반영된다. 도 6은 단지 사용자 인터페이스의 맛을 보이기 위한 것이며, 본 기술 분야에서 잘 알려진 다양한 캐시 분석 도구를 나타내지는 못한다.

<75> 본 발명의 원리가 적용될 수 있는 여러 가지 가능한 실시예의 관점에서, 도면을 참조하여 본 명세서에서 설명한 실시예는 설명을 위한 것일 뿐이며 본 발명의 범위를 제한하고자 하는 것은 아니다. 예컨대 본 기술 분야의 당업자는 본 명세서에서 설명된 실시예, 특히 이에 기초한 데이터 구조 및 프로시저는 본 발명의 사상에서 벗어남이 없이 그 배치나 세부 사항이 수정될 수 있음을 인지하게 될 것이다. 비록 본 발명을 소프트웨어 모듈 또는 구성요소와 관련하여 설명하였지만, 본 기술 분야의 당업자는 이러한 것들이 하드웨어 구성요소로 동등하게 대체될 수 있음을 인지하게 될 것이다. 따라서 본 명세서에서 설명한 본 발명은 이하의 청구 범위 및 그 균등의 범위 내에 포함될 수 있는 모든 실시예를 고려한 것이다.

발명의 효과

<76> 상기한 바에 비추어, 본 발명은 잠금으로 인한 지연 오버헤드 없이 갱신될 수 있고, 캐시 내 데이터의 중요도에 관한 포괄적 기준을 갖는 중앙 캐시를 제공한다. "원자적" 갱신은 록에 의한 지연 오버헤드를 초래하지 않고 액세스 조정을 할 수 있는 이점을 제공한다. 캐시 갱신은 중도에 인터럽트될 수 없도록 설계되었다는 점에서 "원자적"이다. 그 결과, 캐시는 애플리케이션의 액세스 시에 항상 최신이자 일관된 상태가 된다.

도면의 간단한 설명

<1> 도 1a는 LAN(Local Area Network)을 통해 캐시 데이터를 공유하는 세 개의 컴퓨팅 장치를 도시하는 블록도.

<2> 도 1b는 도 1a의 컴퓨팅 장치간에 공유되는 중앙 캐시 메모리를 도시하는 블록도.

<3> 도 2는 본 발명을 지원하는 컴퓨터 시스템의 예를 일반적으로 도시하는 개략도.

<4> 도 3a 내지 도 3d는 본 발명에 따른 중앙 캐시로부터 데이터를 액세스하고자 시도하는 애플리케이션 프로그램을 위한 방법의 예를 도시하는 흐름도.

<5> 도 4a는 본 발명에 따른 중앙 캐시의 데이터 구조의 예를 도시하는 개략도.

<6> 도 4b는 중앙 캐시에 있는 데이터 요소의 데이터 구조의 예를 도시하는 개략도.

<7> 도 5 내지 도 5d는 본 발명에 따라 중앙 캐시를 유지하는 루틴(routine)을 위한 방법의 예를 도시하는 흐름도.

<8> 도 6은 중앙 캐시 서비스 루틴을 구성(configuration)하는 방법의 예를 도시하는 흐름도.

<9> <도면의 주요 부분에 대한 부호의 설명>

<10> 100 : 캐시 데이터 공유 환경

<11> 102 : 컴퓨팅 장치 A

<12> 104 : 컴퓨팅 장치 B

<13> 112 : 애플리케이션 A

<14> 114 : 로컬 및 중앙 캐시 탐색 루틴

<15> 116 : 현재 로컬 캐시

<16> 118 : 현재 중앙 캐시

<17> 120 : 로컬 캐시 갱신 루틴

<18> 122 : 이전 로컬 캐시

<19> 124 : 중앙 캐시 서비스 루틴

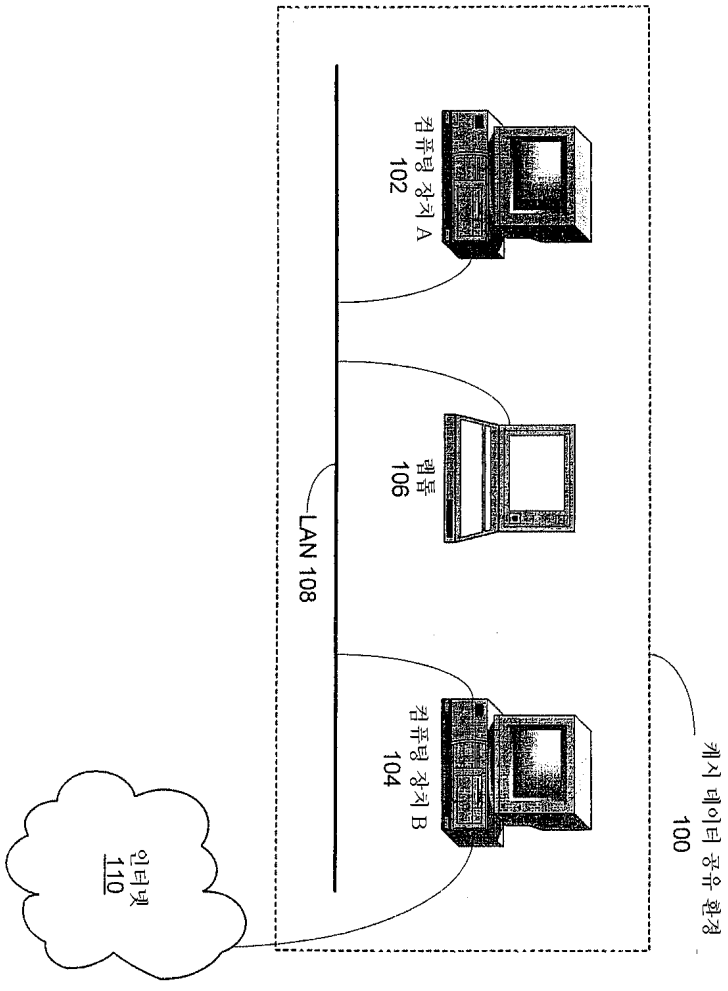
<20> 126 : 이전 중앙 캐시

<21> 128 : 애플리케이션 B

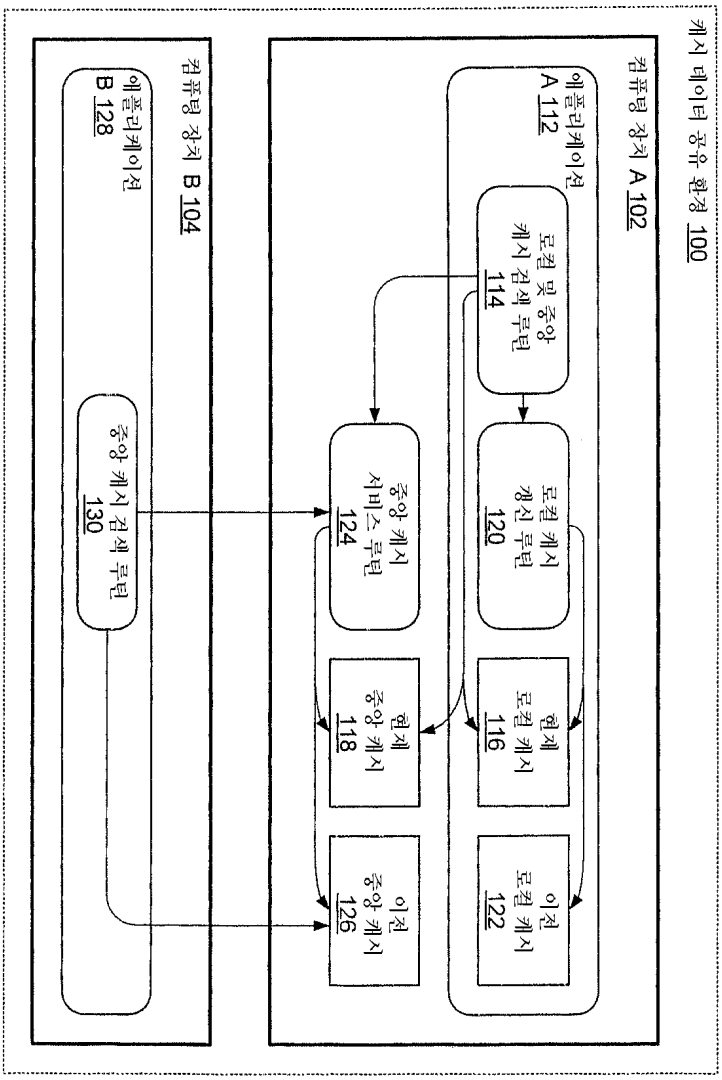
<22> 130 : 중앙 캐시 탐색 루틴

도면

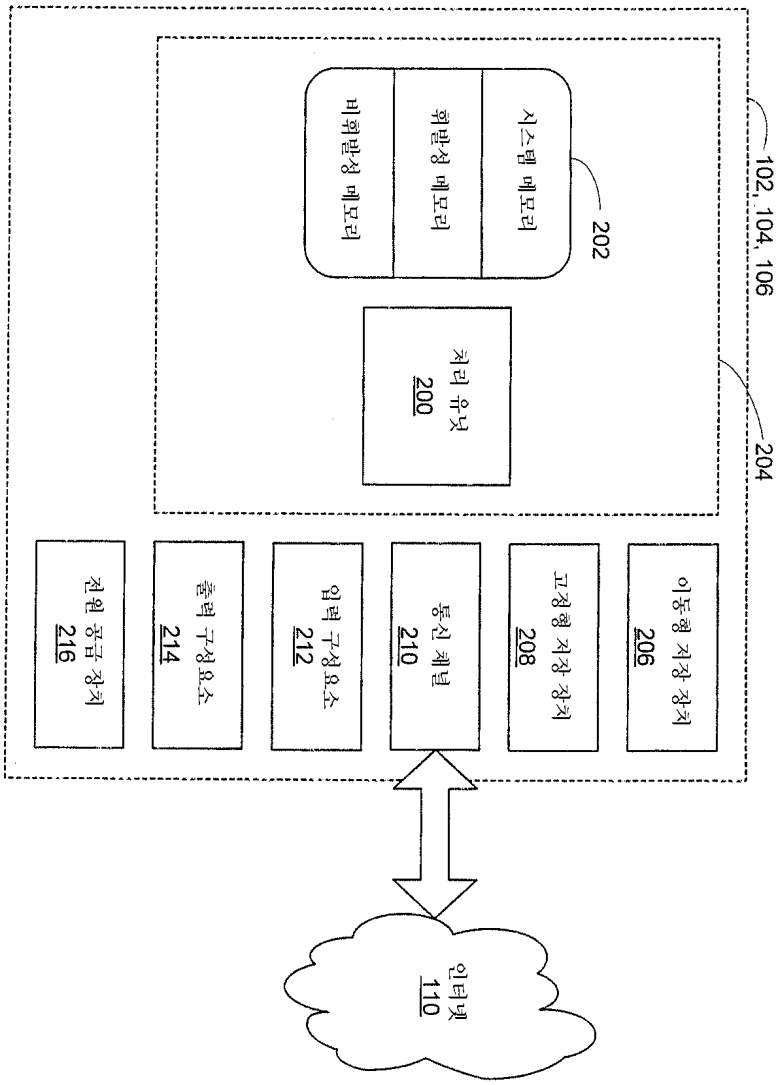
도면1a



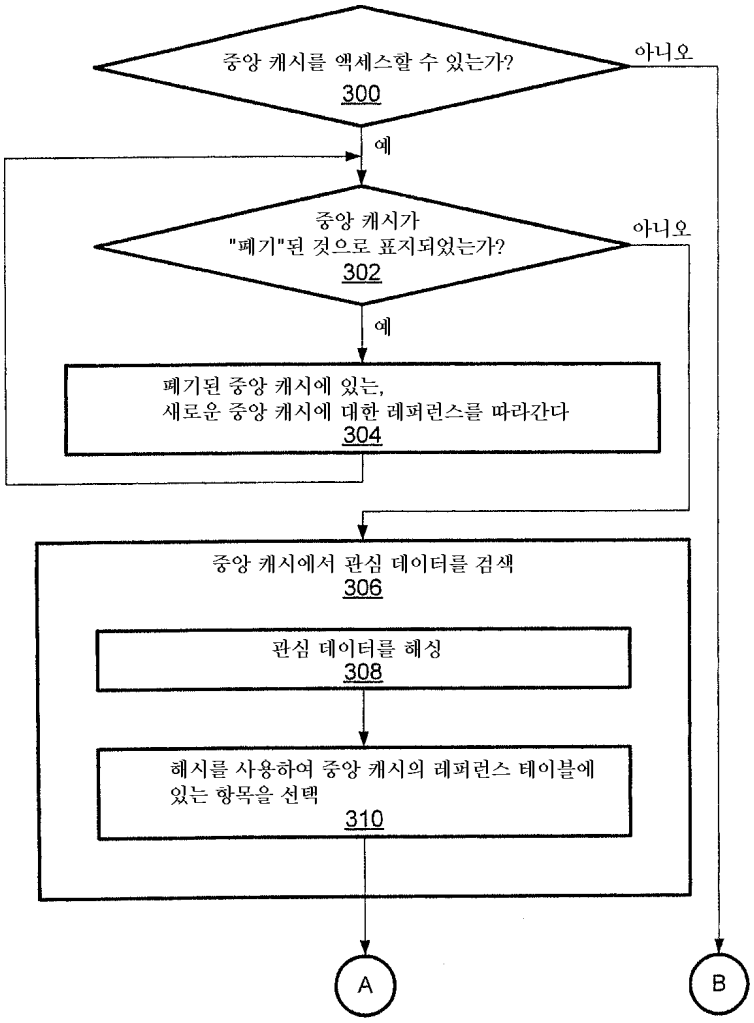
도면1b


캐시 데이터 공유 환경 100
컴퓨팅 장치 A 102
에플리케이션 A 112
로컬 및 중앙 캐시 검색 루틴 114
로컬 캐시 갱신 루틴 120
현재 로컬 캐시 116
이전 로컬 캐시 122
중앙 캐시 서비스 루틴 124
현재 중앙 캐시 118
이전 중앙 캐시 126
컴퓨팅 장치 B 104
에플리케이션 B 128
중앙 캐시 검색 루틴 130

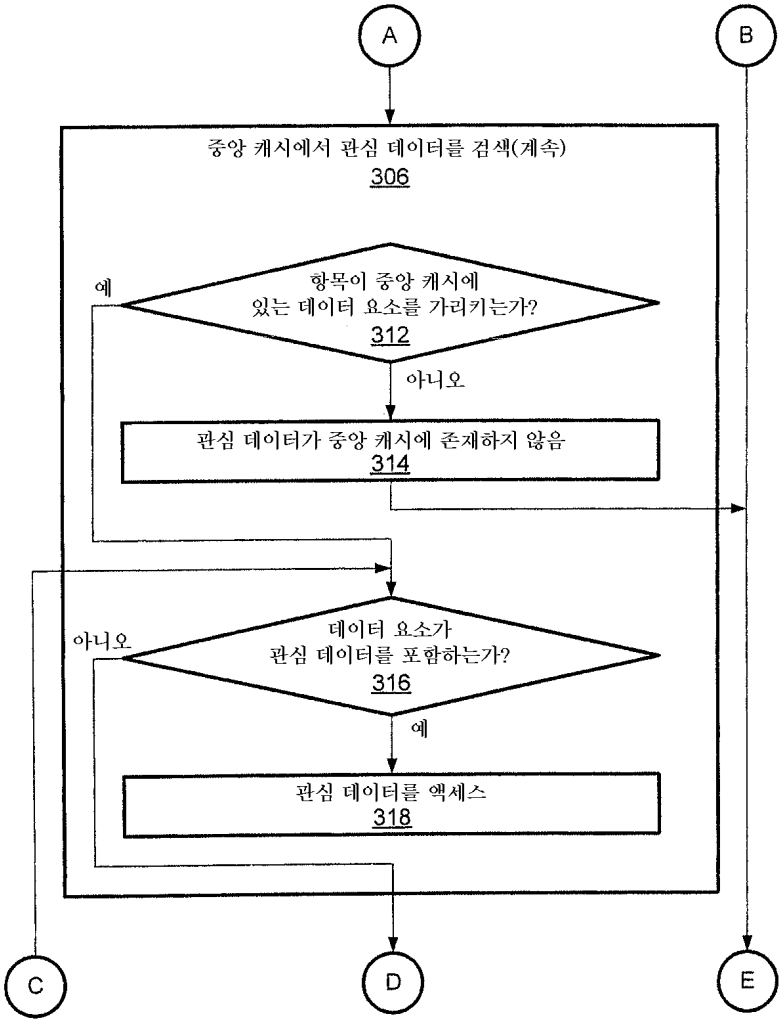
도면2


102, 104, 106
204
202
시스템 메모리
휘발성 메모리
비휘발성 메모리
처리 유닛
200
이동형 저장 장치
206
고정형 저장 장치
208
통신 채널
210
입력 구성요소
212
출력 구성요소
214
전원 공급 장치
216
인터넷
110

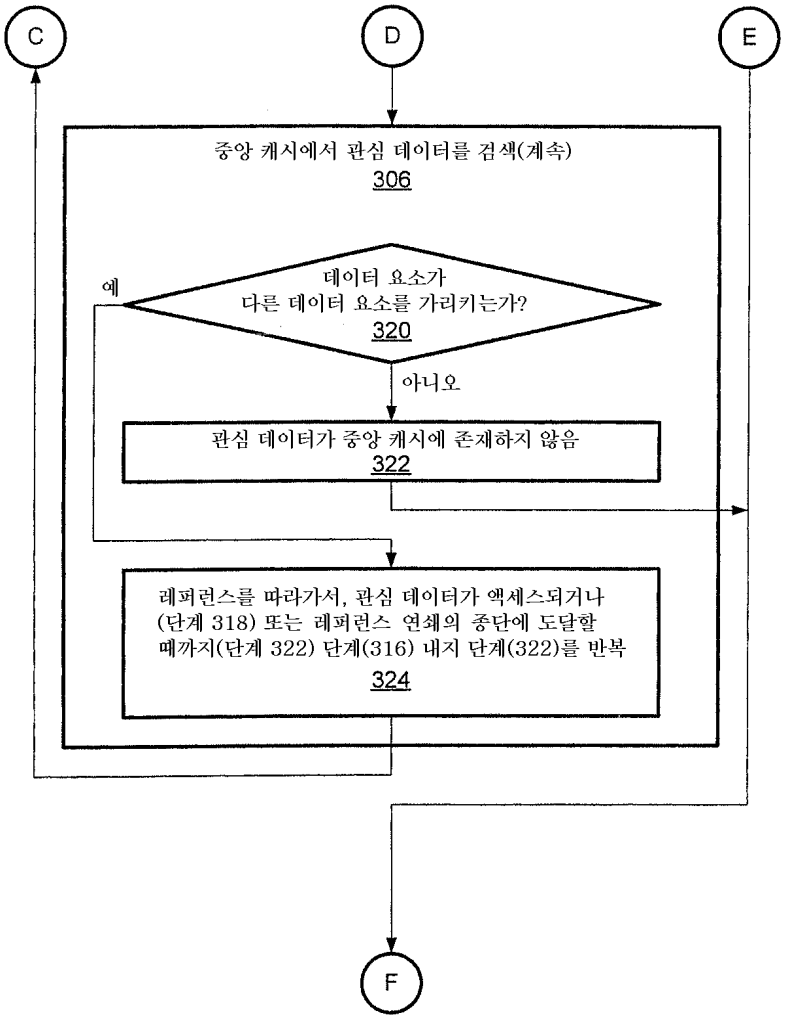
도면3a


중앙 캐시를 액세스할 수 있는가?
300
아니오
예
중앙 캐시가
"폐기"된 것으로 표지되었는가?
302
아니오
예
폐기된 중앙 캐시에 있는,
새로운 중앙 캐시에 대한 레퍼런스를 따라간다
304
중앙 캐시에서 관심 데이터를 검색
306
관심 데이터를 해싱
308
해시를 사용하여 중앙 캐시의 레퍼런스 테이블에
있는 항목을 선택
310
A
B

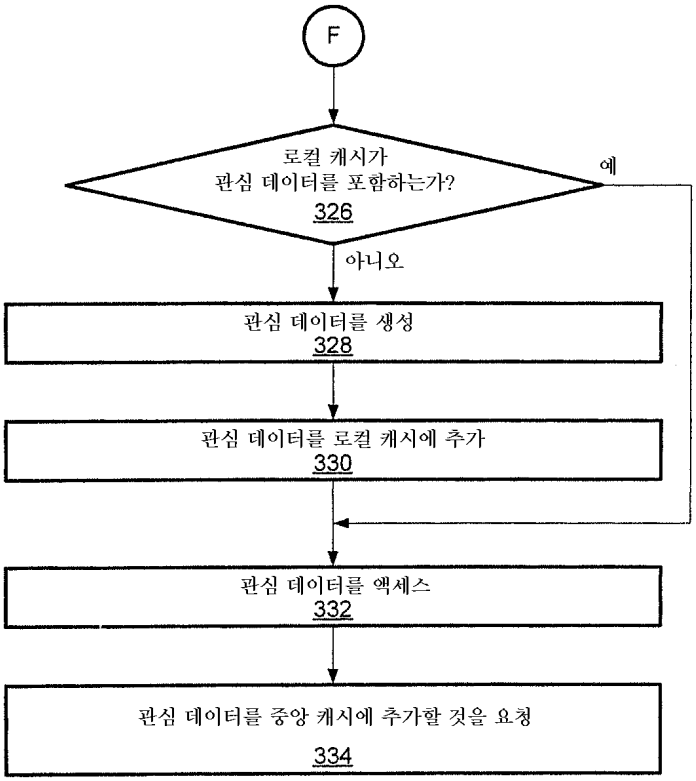
도면3b


A
B
중앙 캐시에서 관심 데이터를 검색(계속)
306
예
항목이 중앙 캐시에
있는 데이터 요소를 가리키는가?
312
아니오
관심 데이터가 중앙 캐시에 존재하지 않음
314
아니오
데이터 요소가
관심 데이터를 포함하는가?
316
예
관심 데이터를 액세스
318
C
D
E

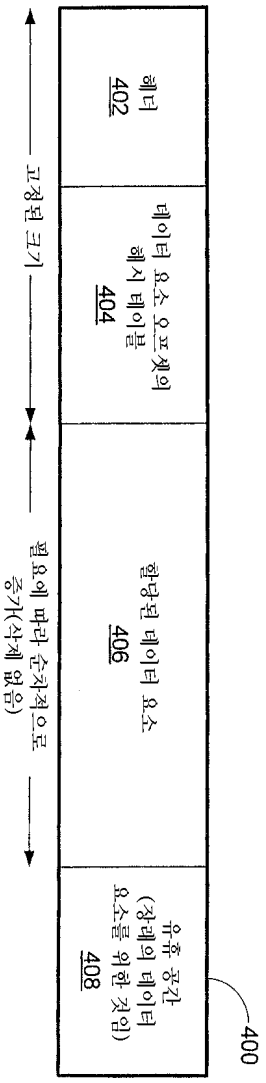
도면3c


C
D
E
중앙 캐시에서 관심 데이터를 검색(계속)
306
데이터 요소가
다른 데이터 요소를 가리키는가?
320
예
아니오
관심 데이터가 중앙 캐시에 존재하지 않음
322
레퍼런스를 따라가서, 관심 데이터가 액세스되거나
(단계 318) 또는 레퍼런스 연쇄의 종단에 도달할
때까지(단계 322) 단계(316) 내지 단계(322)를 반복
324
F

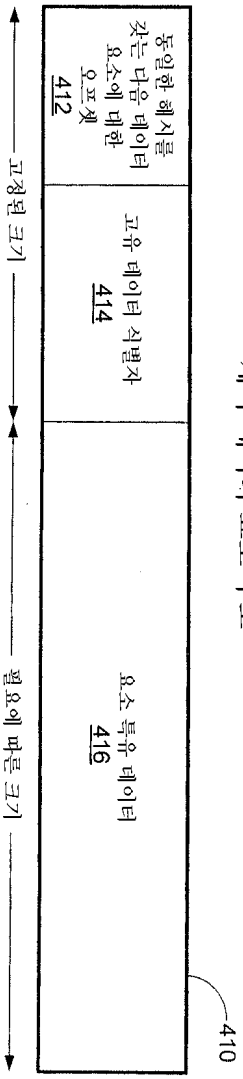
도면3d


F
로컬 캐시가
관심 데이터를 포함하는가?
326
예
아니오
관심 데이터를 생성
328
관심 데이터를 로컬 캐시에 추가
330
관심 데이터를 액세스
332
관심 데이터를 중앙 캐시에 추가할 것을 요청
334

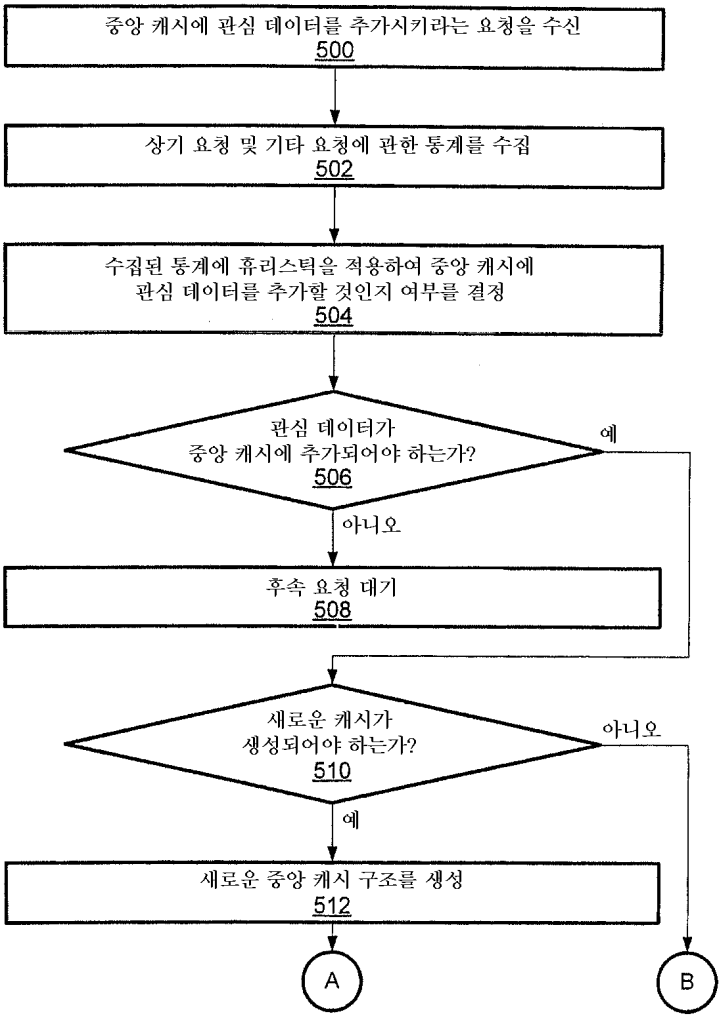
도면4a


중앙 캐시 파일 구조
400
헤더
402
데이터 요소 오프셋의
해시 테이블
404
할당된 데이터 요소
406
유휴 공간
(장래의 데이터
요소를 위한 것임)
408
고정된 크기
필요에 따라 순차적으로
증가(삭제 없음)

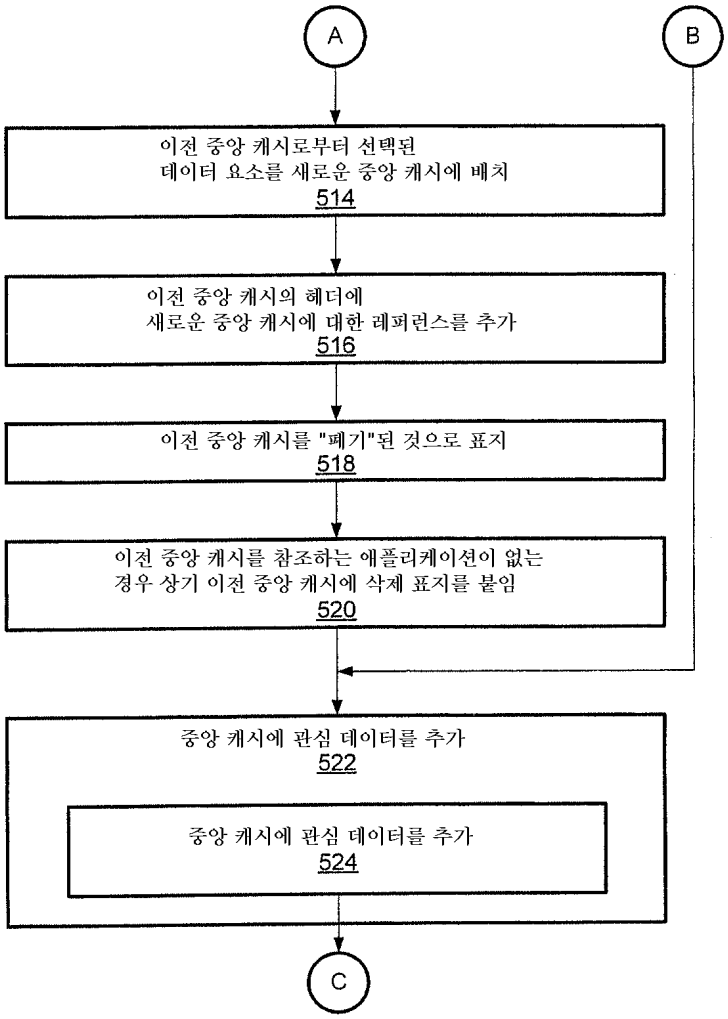
도면4b


캐시 데이터 요소 구조
410
동일한 해시를 갖는 다음 데이터 요소에 대한 오프셋 412
고유 데이터 식별자 414
요소 특유 데이터 416
고정된 크기
필요에 따른 크기

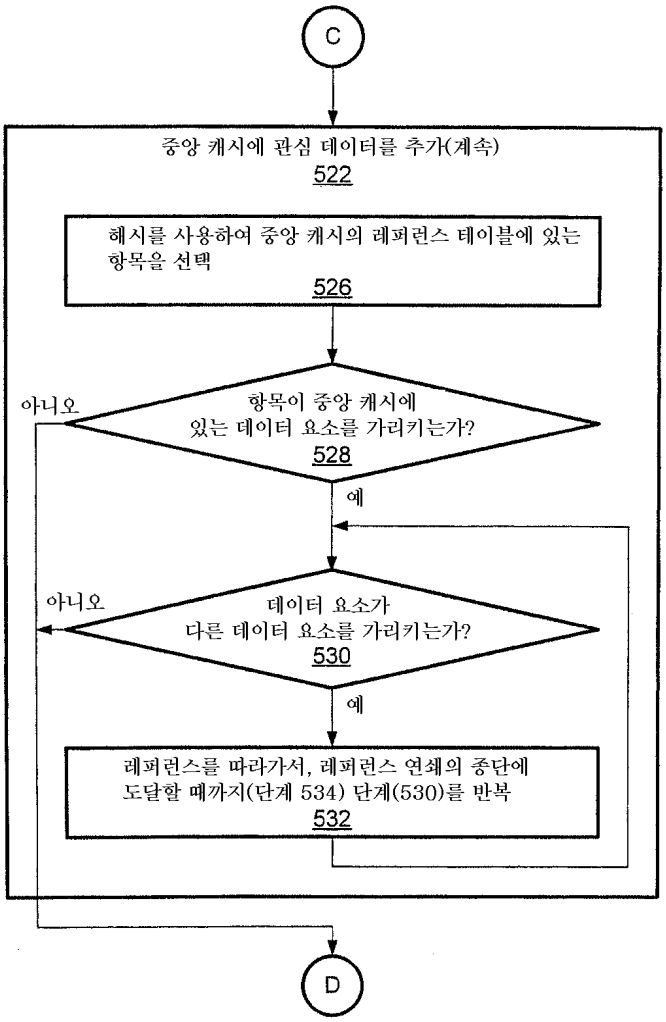
도면5a


중앙 캐시에 관심 데이터를 추가시키라는 요청을 수신
500
상기 요청 및 기타 요청에 관한 통계를 수집
502
수집된 통계에 휴리스틱을 적용하여 중앙 캐시에 관심 데이터를 추가할 것인지 여부를 결정
504
관심 데이터가 중앙 캐시에 추가되어야 하는가?
506
예
아니오
후속 요청 대기
508
새로운 캐시가 생성되어야 하는가?
510
아니오
예
새로운 중앙 캐시 구조를 생성
512
A
B

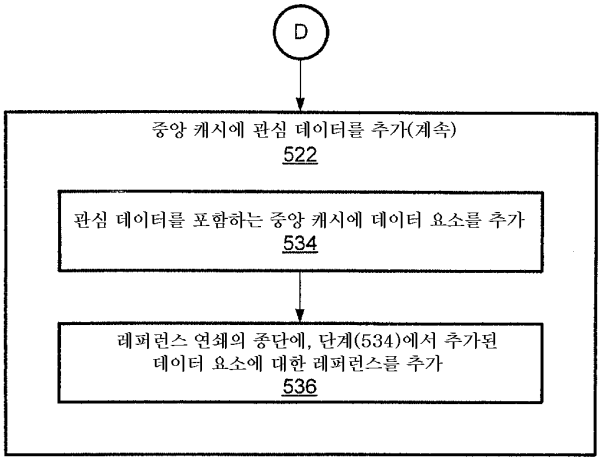
도면5b


A
B
이전 중앙 캐시로부터 선택된
데이터 요소를 새로운 중앙 캐시에 배치
514
이전 중앙 캐시의 헤더에
새로운 중앙 캐시에 대한 레퍼런스를 추가
516
이전 중앙 캐시를 "폐기"된 것으로 표지
518
이전 중앙 캐시를 참조하는 애플리케이션이 없는
경우 상기 이전 중앙 캐시에 삭제 표지를 붙임
520
중앙 캐시에 관심 데이터를 추가
522
중앙 캐시에 관심 데이터를 추가
524
C

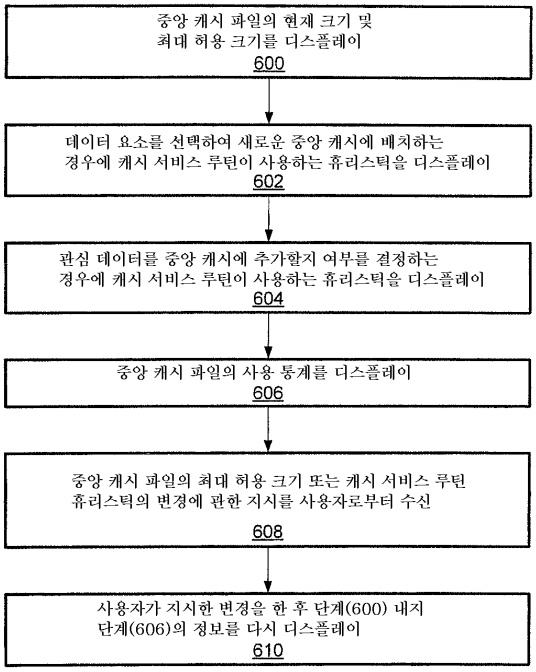
도면5c


C
중앙 캐시에 관심 데이터를 추가(계속)
522
해시를 사용하여 중앙 캐시의 레퍼런스 테이블에 있는 항목을 선택
526
항목이 중앙 캐시에 있는 데이터 요소를 가리키는가?
528
아니오
예
데이터 요소가 다른 데이터 요소를 가리키는가?
530
아니오
예
레퍼런스를 따라가서, 레퍼런스 연쇄의 종단에 도달할 때까지(단계 534) 단계(530)를 반복
532
D

도면5d


D
중앙 캐시에 관심 데이터를 추가(계속)
522
관심 데이터를 포함하는 중앙 캐시에 데이터 요소를 추가
534
레퍼런스 연쇄의 종단에, 단계(534)에서 추가된 데이터 요소에 대한 레퍼런스를 추가
536

도면6


중앙 캐시 파일의 현재 크기 및
최대 허용 크기를 디스플레이
600
데이터 요소를 선택하여 새로운 중앙 캐시에 배치하는
경우에 캐시 서비스 루틴이 사용하는 휴리스틱을 디스플레이
602
관심 데이터를 중앙 캐시에 추가할지 여부를 결정하는
경우에 캐시 서비스 루틴이 사용하는 휴리스틱을 디스플레이
604
중앙 캐시 파일의 사용 통계를 디스플레이
606
중앙 캐시 파일의 최대 허용 크기 또는 캐시 서비스 루틴
휴리스틱의 변경에 관한 지시를 사용자로부터 수신
608
사용자가 지시한 변경을 한 후 단계(600) 내지
단계(606)의 정보를 다시 디스플레이
610